Conjuntos, Lógica, Computação

Uma Introdução Aberta à Metalógica

F25

Conjuntos, Lógica, Computação

The Open Logic Project

Instigador

Richard Zach, *University of Calgary*

Conselho editorial

Aldo Antonelli,† *University of California, Davis*
Andrew Arana, *Université de Lorraine*
Jeremy Avigad, *Carnegie Mellon University*
Tim Button, *University College London*
Walter Dean, *University of Warwick*
Benedict Eastaugh, *University of Warwick*
Gillian Russell, *Australian National University*
Nicole Wyatt, *University of Calgary*
Audrey Yap, *University of Victoria*

Colaboradores

Samara Burns, *Columbia University*
Dana Hägg, *University of Calgary*
Zesen Qian, *Carnegie Mellon University*

Conjuntos, Lógica, Computação

Uma Introdução Aberta à Metalógica

Remixado por Richard Zach

Tradução e adaptação de Pedro Lima

Outono 2025

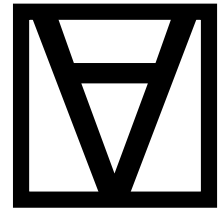
O Open Logic Project gostaria de agradecer o generoso apoio do Taylor Institute for Teaching and Learning da University of Calgary e da Alberta Open Educational Resources (ABOER) Initiative, tornada possível por um investimento do governo de Alberta.

UNIVERSITY OF CALGARY
Taylor Institute for Teaching and Learning

Ilustrações da capa por Matthew Leadbeater, usadas sob uma Licença Creative Commons Atribuição-NãoComercial 4.0 Internacional.

Composto em Baskervald X e Nimbus Sans com LaTeX.

Esta versão de *Conjuntos, Lógica, Computação* é a revisão 51c2271 (2026-07-12), com conteúdo gerado a partir da revisão 51c2271 (2026-07-12) do *Open Logic Text*. Disponível gratuitamente em:
https://slc.openlogicproject.org/pt/



Sumário

Thoralf Skolem
1887 - 1963

Prefácio

Este livro é uma introdução à metalógica, voltada especialmente a estudantes de ciência da computação e de filosofia. «Metalógica» é assim chamada porque é a disciplina que estuda a própria lógica. A lógica propriamente dita trata dos cânones de inferência válida, e sua versão simbólica ou formal apresenta esses cânones usando linguagens formais, como as da lógica proposicional e da lógica de primeira ordem. A metalógica investiga as propriedades dessas linguagens e dos cânones de inferência correta que as utilizam. Ela estuda tópicos como o de dar significado preciso às expressões dessas linguagens formais, como justificar os cânones de inferência válida, quais são as propriedades de vários sistemas de derivação, incluindo suas propriedades computacionais. Essas questões são importantes e interessantes por si mesmas, porque as linguagens e os sistemas de derivação investigados são aplicados em muitas áreas diferentes—em matemática, filosofia, ciência da computação e linguística, especialmente—, mas também servem como exemplos de como estudar sistemas formais em geral. As linguagens lógicas que estudamos aqui não são as únicas de interesse. Por exemplo, linguistas e filósofos se interessam por linguagens muito mais complicadas do que as da lógica proposicional e da lógica de primeira ordem, e cientistas da computação se interessam por *outros tipos* de linguagens, como linguagens de programação. E os métodos que discutimos—como dar semântica a linguagens formais, como provar resultados sobre

linguagens formais, como investigar as propriedades de linguagens formais—também são aplicáveis nesses casos.

Como qualquer disciplina, a metalógica tem um conjunto de resultados ou fatos e um repertório de métodos e técnicas, e este texto cobre ambos. Muitos estudantes não precisarão conhecer todos os resultados que discutimos fora deste curso, mas precisarão e usarão os métodos que empregamos para estabelecê-los. O teorema de Löwenheim-Skolem, por exemplo, não aparece com frequência em ciência da computação, mas os métodos que usamos para prová-lo sim. Por outro lado, muitos dos resultados que discutimos têm relevância para certos debates, digamos, na filosofia da ciência e na metafísica. Estudantes de filosofia podem não precisar ser capazes de provar esses resultados fora deste curso, mas precisam entender quais são os resultados—e você só *entende* de fato esses resultados se refletiu sobre as definições e provas necessárias para estabelecê-los. Essas são, em parte, as razões pelas quais os resultados e os métodos cobertos neste texto são estudo recomendado—em alguns casos até obrigatório—para estudantes de ciência da computação e de filosofia.

O material está dividido em três partes. A parte I trata da teoria dos conjuntos. Lógica e metalógica estão historicamente muito ligadas ao que se chama «fundamentos da matemática». Os fundamentos matemáticos tratam de como, em última instância, objetos matemáticos como inteiros, racionais e reais, funções, espaços etc. devem ser entendidos. A teoria dos conjuntos fornece uma resposta (há outras), e por isso teoria dos conjuntos e lógica há muito são estudadas em paralelo. Conjuntos, relações e funções também são onipresentes em qualquer investigação formal, não só em matemática, mas também em ciência da computação e em alguns cantos mais técnicos da filosofia. Certamente, para formular e provar resultados sobre semântica e teoria da prova da lógica e sobre os fundamentos da computabilidade, é essencial dispor de uma terminologia para isso. Por exemplo, falaremos de conjuntos de expressões, relações de consequência e de demonstrabilidade, interpretações de símbolos de predicado

(que se revelam relações), funções computáveis e várias relações e construções que as envolvem. Será bom ter símbolos abreviados para isso e refletir sobre as propriedades gerais de conjuntos, relações e funções. Se você não está acostumado a pensar matematicamente e a formular provas matemáticas, pense na primeira parte sobre teoria dos conjuntos como um campo de treino: todas as definições básicas serão dadas, e apresentaremos provas cada vez mais complicadas usando-as. Note que entender essas provas—e ser capaz de encontrá-las e formulá-las você mesmo—é talvez mais importante do que entender os resultados, especialmente na primeira parte. Se o pensamento matemático é novo para você, é importante que você reflita sobre os exemplos e problemas.

Na primeira parte estabeleceremos um resultado importante, no entanto. Esse resultado—o teorema de Cantor—depende de um dos exemplos mais marcantes de análise conceitual que se encontra nas ciências, a saber, a análise do infinito por Cantor. O infinito intrigou matemáticos e filósofos por séculos. Até Cantor, ninguém sabia como pensar adequadamente sobre ele. Muitas pessoas chegaram a considerar um erro pensar sobre o infinito e acreditavam que a noção de coleção infinita em si era incoerente. Cantor transformou o infinito em um assunto com o qual podemos trabalhar de forma coerente e desenvolveu toda uma teoria de coleções infinitas—e números infinitos com os quais podemos medir os tamanhos de coleções infinitas. Ele mostrou que há diferentes níveis de infinito. Essa teoria de números «transfinitos» é bela e intricada, e não nos aprofundaremos muito nela; mas seremos capazes de mostrar que há diferentes níveis de infinito, especificamente, que há níveis «contáveis» e «incontáveis» de infinito. Esse resultado tem aplicações importantes, mas também é realmente o tipo de resultado que qualquer matemático, cientista da computação e filósofo que se preze deveria conhecer.

Na parte II, passamos à lógica de primeira ordem. Definiremos a linguagem da lógica de primeira ordem e sua semântica, isto é, o que são estruturas de primeira ordem e quando uma sentença da lógica de primeira ordem é verdadeira numa estrutura.

Isso nos permitirá fazer duas coisas importantes: (1) podemos definir, com precisão matemática, quando uma sentença é consequência lógica de outra; (2) também podemos considerar como as relações que compõem uma estrutura de primeira ordem são descritas—caracterizadas—pelas sentenças que são verdadeiras nelas. Isso nos leva, em particular, a uma discussão do método axiomático, no qual sentenças de linguagens de primeira ordem são usadas para caracterizar certos tipos de estruturas. A teoria da prova nos ocupará em seguida, e consideraremos a versão original do cálculo de sequentes e da dedução natural definidas na década de 1930 por Gerhard Gentzen. (Seu professor pode optar por cobrir apenas um deles; então qualquer referência a «derivações» e «derivabilidade» significará o sistema que ele escolheu.) A noção semântica de consequência e a noção sintática de derivabilidade nos dão duas maneiras completamente diferentes de tornar precisa a ideia de que uma sentença pode seguir de outras. Os teoremas de correção e completude ligam essas duas caracterizações. Em particular, provaremos o teorema da completude de Gödel, que afirma que sempre que uma sentença é consequência semântica de outras, ela também é derivável a partir delas. Uma formulação equivalente é: se uma coleção de sentenças é consistente—no sentido de que nada contraditório pode ser provado a partir delas—, então existe uma estrutura que torna todas elas verdadeiras.

A segunda formulação do teorema da completude é talvez a mais surpreendente. Por volta da época em que Gödel provou esse resultado (em 1929), o matemático alemão David Hilbert defendia famosamente a visão de que consistência (isto é, ausência de contradição) é tudo o que a existência matemática exige. Em outras palavras, sempre que um matemático pode descrever coerentemente uma estrutura ou classe de estruturas, ele deveria ter o direito de acreditar na existência dessas estruturas. Na época, muitos achavam essa ideia absurda: só porque você pode descrever uma estrutura sem se contradizer, certamente não se segue que tal estrutura realmente existe. Mas é exatamente isso que o teorema da completude de Gödel diz. Além desse aspecto

paradoxal—e certamente intrigante do ponto de vista filosófico—, o teorema da completude também tem duas aplicações importantes que nos permitem provar outros resultados sobre a existência de estruturas que tornam certas sentenças verdadeiras. São os teoremas da compacidade e de Löwenheim-Skolem.

Na parte III, conectamos lógica e computabilidade. De novo, há uma conexão histórica: David Hilbert havia colocado como problema fundamental da lógica encontrar um método mecânico que decidisse, para uma dada sentença da lógica, se ela tem uma prova. Tal método existe, é claro, para a lógica proposicional: basta verificar todas as tabelas-verdade, e como há apenas um número finito delas, o método eventualmente produz uma resposta correta. Um método tão direto não é possível para a lógica de primeira ordem, já que o número de estruturas possíveis é infinito (e as próprias estruturas podem ser infinitas). Lógicos trabalharam por anos para encontrar métodos mais engenhosos. Alonzo Church e Alan Turing acabaram estabelecendo que não existe tal método. Para fazer isso, foi necessário primeiro fornecer uma definição precisa do que é, em geral, um método mecânico. Se um procedimento de decisão tivesse sido proposto, presumivelmente ele teria sido reconhecido como um método efetivo. Para provar que nenhum método efetivo existe, é preciso definir «método efetivo» primeiro e dar uma prova de impossibilidade com base nessa definição. Foi isso que Turing fez: ele propôs a ideia de uma máquina de Turing[1] como modelo matemático do que um procedimento mecânico pode, em princípio, fazer. Esse é outro exemplo de *análise conceitual* de um conceito informal usando maquinaria matemática; e talvez seja de importância comparável para a ciência da computação à análise do infinito por Cantor para a matemática. Nossa última grande empreitada será a prova de dois teoremas de impossibilidade: mostraremos que o chamado «problema da parada» não pode ser resolvido por máquinas de Turing e, por fim, que o «problema da decisão» de Hilbert (para a lógica) também não pode.

[1] Turing, é claro, não a chamou assim.

Este texto é matemático, no sentido de que discutimos definições matemáticas e provamos nossos resultados matematicamente. Mas não é matemático no sentido de que você precise de amplo conhecimento prévio de matemática. Nada neste texto exige conhecimento de álgebra, trigonometria ou cálculo. Fizemos um esforço especial para também não exigir familiaridade com o modo de trabalho da matemática: de fato, parte do objetivo é *desenvolver* os tipos de raciocínio e habilidades de prova necessários para entender e provar nossos resultados. A organização do texto segue a convenção matemática, por um motivo: essas convenções foram desenvolvidas porque clareza e precisão são especialmente importantes, e assim, por exemplo, é crucial saber quando algo é afirmado como conclusão de um argumento, é oferecido como razão para outra coisa ou pretende introduzir novo vocabulário. Por isso seguimos a convenção matemática e rotulamos passagens como «definições» se servem para introduzir nova terminologia ou símbolos; e como «teoremas», «proposições», «lemas» ou «corolários» quando registramos um resultado ou achado. Fora essas convenções, usaremos os métodos de prova lógica que talvez já sejam familiares de um primeiro curso de lógica, e também faremos amplo uso do método de *indução* para provar resultados. Dois capítulos do apêndice são dedicados a esses métodos de prova.

Notas para instrutores O material deste livro é adequado para um segundo curso semestral de lógica formal. Eu o cubro em 12 semanas em Lógica II na University of Calgary, embora não cubra tudo com o mesmo nível de detalhe que há neste livro. Por exemplo, normalmente trato apenas de dedução natural e deixo de fora provas detalhadas de completude para identidade. Os estudantes já cursaram Lógica I, normalmente a partir de *forall x: Calgary*, que usa as mesmas regras de dedução natural, exceto no formato Fitch.

A versão mais recente deste livro está disponível em PDF em slc.openlogicproject.org, mas muda com frequência. A licença

CC BY lhe dá o direito de baixar e distribuir o livro você mesmo. Para garantir que todos os seus estudantes tenham a mesma versão do livro durante o semestre em que o usam, você deve fazê-lo: envie o PDF que decidir usar para o seu LMS em vez de apenas dar aos estudantes o link. Você também pode mandar imprimir os PDFs na livraria da universidade, mas algumas livrarias poderão comprar e estocar os livros de capa mole disponíveis na Amazon.

A sintaxe, a semântica e os sistemas de prova da lógica de primeira ordem são apoiados pelo aplicativo gratuito e online de ensino de lógica *Carnap*, de Graham Leach-Krouse (carnap.io). Isso permite envio e correção automática de exercícios como derivações de dedução natural e de cálculo de sequentes, fornecimento de estruturas para teorias simples e exercícios de simbolização. Há também um simulador de máquinas de Turing em turing.openlogicproject.org que pode ser usado para ilustrar o material da parte III. Os exemplos lá estão disponíveis pré-carregados no simulador.

Georg Cantor
1845 - 1918

PARTE I

Conjuntos, Relações e Funções

CAPÍTULO 1

Conjuntos

1.1 Extensionalidade

Um *conjunto* é uma coleção de objetos, considerada como um único objeto. Os objetos que compõem o conjunto chamam-se *elementos* ou *membros* do conjunto. Se x é um elemento de um conjunto A, escrevemos $x \in A$; caso contrário, escrevemos $x \notin A$. O conjunto que não tem elementos chama-se o *conjunto vazio* e denota-se "$\emptyset$".

Não importa como *especificamos* o conjunto, ou como *ordenamos* os seus elementos, ou mesmo quantas *vezes* contamos os seus elementos. Só importa quais são os seus elementos. Codificamos isto no princípio seguinte.

Definição 1.1 (Extensionalidade). Se A e B são conjuntos, então $A = B$ se e só se todo elemento de A é também um elemento de B, e vice-versa.

A extensionalidade autoriza certa notação. Em geral, quando temos objetos a_1, ..., a_n, então $\{a_1, \ldots, a_n\}$ é *o* conjunto cujos elementos são $a_1, \ldots, a_n$. Enfatizamos a palavra "*o*", pois a extensionalidade nos diz que só pode haver *um* tal conjunto. De fato, a extensionalidade também autoriza o seguinte:

$$\{a, a, b\} = \{a, b\} = \{b, a\}.$$

Isto traduz a ideia de que, quando consideramos conjuntos, não nos importa a ordem dos seus elementos, nem quantas vezes são especificados.

Exemplo 1.2. Sempre que temos vários objetos, podemos reuni-los num conjunto. O conjunto dos irmãos de Richard, por exemplo, é um conjunto que contém uma pessoa, e podemos escrevê-lo como $S = \{\text{Ruth}\}$. O conjunto dos inteiros positivos menores que 4 é $\{1,2,3\}$, mas também se pode escrever como $\{3,2,1\}$ ou mesmo como $\{1,2,1,2,3\}$. São todos o mesmo conjunto, pela extensionalidade. De fato, todo elemento de $\{1,2,3\}$ é também um elemento de $\{3,2,1\}$ (e de $\{1,2,1,2,3\}$), e vice-versa.

Com frequência especificaremos um conjunto por alguma propriedade que os seus elementos compartilham. Usaremos a seguinte notação abreviada para isso: $\{x : \varphi(x)\}$, onde $\varphi(x)$ representa a propriedade que x tem de satisfazer para ser contado entre os elementos do conjunto.

Exemplo 1.3. No nosso exemplo, também poderíamos ter especificado S como

$$S = \{x : x \text{ é irmão de Richard}\}.$$

Exemplo 1.4. Diz-se que um número é *perfeito* se e só se for igual à soma dos seus divisores próprios (isto é, números que o dividem sem resto mas não são iguais ao próprio número). Por exemplo, 6 é perfeito porque os seus divisores próprios são 1, 2, e 3, e $6 = 1+2+3$. De fato, 6 é o único inteiro positivo menor que 10 que é perfeito. Assim, usando a extensionalidade, podemos dizer:

$$\{6\} = \{x : x \text{ é perfeito e } 0 \leq x \leq 10\}$$

Lemos a notação à direita como "o conjunto dos x tais que x é perfeito e $0 \leq x \leq 10$". A identidade aqui confirma que, quando consideramos conjuntos, não nos importa como são especificados. E, mais em geral, a extensionalidade garante que há sempre apenas um conjunto dos x tais que $\varphi(x)$. Logo, a extensionalidade justifica chamar a $\{x : \varphi(x)\}$ *o* conjunto dos x tais que $\varphi(x)$.

A extensionalidade nos dá um modo de mostrar que conjuntos são idênticos: para mostrar que $A = B$, mostre que sempre que $x \in A$ então também $x \in B$, e sempre que $y \in B$ então também $y \in A$.

1.2 Subconjuntos e conjuntos potência

Com frequência quereremos comparar conjuntos. E um tipo óbvio de comparação que se pode fazer é o seguinte: *tudo o que está num conjunto está também no outro.* Esta situação é suficientemente importante para introduzirmos nova notação.

Definição 1.5 (Subconjunto). Se todo elemento de um conjunto A é também um elemento de B, então dizemos que A é um *subconjunto* de B, e escrevemos $A \subseteq B$. Se A não é subconjunto de B, escrevemos $A \nsubseteq B$. Se $A \subseteq B$ mas $A \neq B$, escrevemos $A \subsetneq B$ e dizemos que A é um *subconjunto próprio* de B.

Exemplo 1.6. Todo conjunto é subconjunto de si próprio, e $\emptyset$ é subconjunto de todo conjunto. O conjunto dos números naturais pares é subconjunto do conjunto dos números naturais. Também $\{a, b\} \subseteq \{a, b, c\}$. Mas $\{a, b, e\}$ não é subconjunto de $\{a, b, c\}$.

Exemplo 1.7. O número 2 é um elemento do conjunto dos inteiros, ao passo que o conjunto dos números pares é subconjunto do conjunto dos inteiros. Contudo, um conjunto pode tanto ser um elemento quanto subconjunto de algum outro conjunto, por exemplo, $\{0\} \in \{0, \{0\}\}$ e também $\{0\} \subseteq \{0, \{0\}\}$.

A extensionalidade dá um critério de identidade para conjuntos: $A = B$ se e só se todo elemento de A é também um elemento de B e vice-versa. A definição de "subconjunto" define $A \subseteq B$ precisamente como a primeira metade deste critério: todo elemento de A é também um elemento de B. Claro que a definição também se aplica se trocarmos A e B: isto é, $B \subseteq A$ se e só se todo elemento de B é também um elemento de A. E isso, por sua vez, é

exatamente a parte "vice-versa" da extensionalidade. Em outras palavras, a extensionalidade implica que conjuntos são iguais se e só se são subconjuntos um do outro.

Proposição 1.8. *$A = B$ se e só se $A \subseteq B$ e $B \subseteq A$.*

Agora é também boa ocasião para introduzir mais alguma notação útil. Ao definir quando A é subconjunto de B dissemos que "todo elemento de A é ...," e preenchemos os "..." com "um elemento de B". Mas esta é uma *forma* tão comum de expressão que convém introduzir notação formal para ela.

Definição 1.9. $(\forall x \in A)\varphi$ abrevia $\forall x(x \in A \to \varphi)$. Do mesmo modo, $(\exists x \in A)\varphi$ abrevia $\exists x(x \in A \wedge \varphi)$.

Com esta notação, podemos dizer que $A \subseteq B$ se e só se $(\forall x \in A)x \in B$.

Passamos agora a considerar um certo tipo de conjunto: o conjunto de todos os subconjuntos de um dado conjunto.

Definição 1.10 (Conjunto potência). O conjunto constituído por todos os subconjuntos de um conjunto A chama-se o *conjunto potência* de A, e denota-se $\wp(A)$.

$$\wp(A) = \{B : B \subseteq A\}$$

Exemplo 1.11. Quais são todos os subconjuntos possíveis de $\{a,b,c\}$? São: $\emptyset$, $\{a\}$, $\{b\}$, $\{c\}$, $\{a,b\}$, $\{a,c\}$, $\{b,c\}$, $\{a,b,c\}$. O conjunto de todos estes subconjuntos é $\wp(\{a,b,c\})$:

$$\wp(\{a,b,c\}) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a,b\}, \{b,c\}, \{a,c\}, \{a,b,c\}\}$$

1.3 Alguns Conjuntos Importantes

Exemplo 1.12. Trabalharemos sobretudo com conjuntos cujos elementos são objetos matemáticos. Quatro desses conjuntos são

tão importantes que têm nomes específicos:

$$\mathbb{N} = \{0,1,2,3,\ldots\}$$
o conjunto dos números naturais

$$\mathbb{Z} = \{\ldots,-2,-1,0,1,2,\ldots\}$$
o conjunto dos números inteiros

$$\mathbb{Q} = \{m/n : m,n \in \mathbb{Z} \text{ e } n \neq 0\}$$
o conjunto dos números racionais

$$\mathbb{R} = (-\infty,\infty)$$
o conjunto dos números reais (o contínuo)

São todos conjuntos *infinitos*, isto é, cada um tem infinitos elementos.

À medida que percorremos estes conjuntos, vamos acrescentando *mais* números ao nosso repertório. De fato, deve ficar claro que $\mathbb{N} \subseteq \mathbb{Z} \subseteq \mathbb{Q} \subseteq \mathbb{R}$: afinal, todo número natural é um inteiro; todo inteiro é racional; e todo racional é real. Igualmente, deve ficar claro que $\mathbb{N} \subsetneq \mathbb{Z} \subsetneq \mathbb{Q}$, pois -1 é inteiro mas não natural, e $1/2$ é racional mas não inteiro. É menos óbvio que $\mathbb{Q} \subsetneq \mathbb{R}$, isto é, que há números reais que não são racionais.

Também usaremos por vezes o conjunto dos inteiros positivos $\mathbb{Z}^+ = \{1,2,3,\ldots\}$ e o conjunto que contém apenas os dois primeiros naturais $\mathbb{B} = \{0,1\}$.

Exemplo 1.13 (Cadeias). Outro exemplo interessante é o conjunto A^* das *cadeias finitas* sobre um alfabeto A: qualquer sequência finita de elementos de A é uma cadeia sobre A. Incluímos a *cadeia vazia* Λ entre as cadeias sobre A, para todo alfabeto A. Por exemplo,

$$\mathbb{B}^* = \{\Lambda,0,1,00,01,10,11,\\ 000,001,010,011,100,101,110,111,0000,\ldots\}.$$

Se $x = x_1 \ldots x_n \in A^*$ é uma cadeia constituída por n "letras" de A, então dizemos que o *comprimento* da cadeia é n e escrevemos $\text{len}(x) = n$.

Exemplo 1.14 (Sequências infinitas). Para qualquer conjunto A podemos também considerar o conjunto A^ω das sequências infinitas de elementos de A. Uma sequência infinita $a_1a_2a_3a_4\ldots$ consiste numa lista infinita numa só direção de objetos, cada um dos quais é um elemento de A.

1.4 Uniões e Interseções

Na seção 1.1, introduzimos definições de conjuntos por abstração, isto é, definições da forma $\{x : \varphi(x)\}$. Aqui invocamos alguma propriedade φ, e essa propriedade pode mencionar conjuntos que já definimos. Assim, por exemplo, se A e B são conjuntos, o conjunto $\{x : x \in A \vee x \in B\}$ consiste em todos aqueles objetos que são elementos de A ou de B, isto é, é o conjunto que reúne os elementos de A e B. Podemos visualizar isto como na Figura 1.1, onde a área realçada indica os elementos dos dois conjuntos A e B juntos.

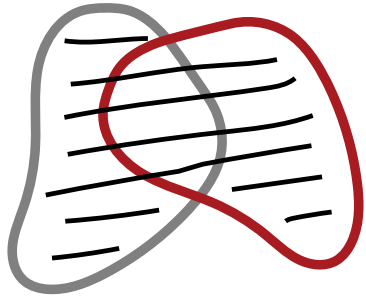


Figura 1.1: A união $A \cup B$ de dois conjuntos é o conjunto dos elementos de A juntamente com os de B.

Esta operação sobre conjuntos—combiná-los—é muito útil e comum, e por isso damos-lhe um nome formal e um símbolo.

Definição 1.15 (União). A *união* de dois conjuntos A e B, escrita $A \cup B$, é o conjunto de todas as coisas que são elementos de

A, de B, ou de ambos.

$$A \cup B = \{x : x \in A \vee x \in B\}$$

Exemplo 1.16. Como a multiplicidade de elementos não importa, a união de dois conjuntos que têm um elemento em comum contém esse elemento apenas uma vez, por exemplo, $\{a,b,c\} \cup \{a,0,1\} = \{a,b,c,0,1\}$.

A união de um conjunto com um dos seus subconjuntos é simplesmente o conjunto maior: $\{a,b,c\} \cup \{a\} = \{a,b,c\}$.

A união de um conjunto com o conjunto vazio é idêntica ao conjunto: $\{a,b,c\} \cup \emptyset = \{a,b,c\}$.

Também podemos considerar uma operação "dual" à união. É a operação que forma o conjunto de todos os elementos que são elementos de A e são também elementos de B. Esta operação chama-se *interseção*, e pode representar-se como na Figura 1.2.

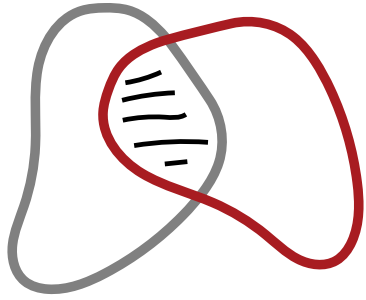


Figura 1.2: A interseção $A \cap B$ de dois conjuntos é o conjunto dos elementos que têm em comum.

Definição 1.17 (Interseção). A *interseção* de dois conjuntos A e B, escrita $A \cap B$, é o conjunto de todas as coisas que são elementos de ambos A e B.

$$A \cap B = \{x : x \in A \wedge x \in B\}$$

Dois conjuntos dizem-se *disjuntos* se a sua interseção é vazia. Isso significa que não têm elementos em comum.

Exemplo 1.18. Se dois conjuntos não têm elementos em comum, a sua interseção é vazia: $\{a,b,c\} \cap \{0,1\} = \emptyset$.

Se dois conjuntos têm elementos em comum, a sua interseção é o conjunto de todos esses elementos: $\{a,b,c\} \cap \{a,b,d\} = \{a,b\}$.

A interseção de um conjunto com um dos seus subconjuntos é simplesmente o conjunto menor: $\{a,b,c\} \cap \{a,b\} = \{a,b\}$.

A interseção de qualquer conjunto com o conjunto vazio é vazia: $\{a,b,c\} \cap \emptyset = \emptyset$.

Também podemos formar a união ou a interseção de mais de dois conjuntos. Uma forma elegante de tratar disso em geral é a seguinte: suponhamos que reunimos num único conjunto todos os conjuntos dos quais queremos formar a união (ou a interseção). Então podemos definir a união de todos os nossos conjuntos originais como o conjunto de todos os objetos que pertencem a pelo menos um elemento desse conjunto, e a interseção como o conjunto de todos os objetos que pertencem a todo elemento desse conjunto.

Definição 1.19. Se A é um conjunto de conjuntos, então $\bigcup A$ é o conjunto dos elementos dos elementos de A:

$$\begin{aligned}\bigcup A &= \{x : x \text{ pertence a um elemento de } A\}, \text{ isto é,}\\ &= \{x : \text{existe um } B \in A \text{ tal que } x \in B\}\end{aligned}$$

Definição 1.20. Se A é um conjunto de conjuntos, então $\bigcap A$ é o conjunto dos objetos que todos os elementos de A têm em

comum:

$$\bigcap A = \{x : x \text{ pertence a todo elemento de } A\}, \text{ isto é,}$$
$$= \{x : \text{para todo } B \in A, x \in B\}$$

Exemplo 1.21. Suponha $A = \{\{a,b\},\{a,d,e\},\{a,d\}\}$. Então $\bigcup A = \{a,b,d,e\}$ e $\bigcap A = \{a\}$.

Também poderíamos fazer o mesmo para uma sequência de conjuntos $A_1, A_2, \ldots$

$$\bigcup_i A_i = \{x : x \text{ pertence a um dos } A_i\}$$
$$\bigcap_i A_i = \{x : x \text{ pertence a todos os } A_i\}.$$

Quando temos um *índice* de conjuntos, isto é, algum conjunto I tal que consideramos A_i para cada $i \in I$, podemos também usar estas abreviaturas:

$$\bigcup_{i \in I} A_i = \bigcup \{A_i : i \in I\}$$
$$\bigcap_{i \in I} A_i = \bigcap \{A_i : i \in I\}$$

Finalmente, podemos querer pensar no conjunto de todos os elementos em A que não estão em B. Podemos representá-lo como na Figura 1.3.

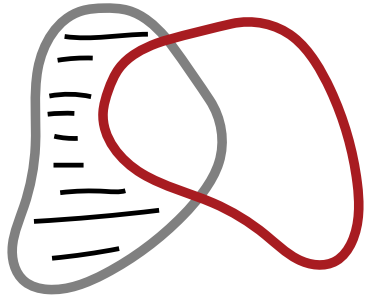


Figura 1.3: A diferença $A \setminus B$ de dois conjuntos é o conjunto dos elementos de A que não são também elementos de B.

Definição 1.22 (Diferença). A *diferença* de conjuntos $A \setminus B$ é o conjunto de todos os elementos de A que não são também elementos de B, isto é,

$$A \setminus B = \{x : x \in A \text{ e } x \notin B\}.$$

1.5 Pares, Tuplas, Produtos Cartesianos

Da extensionalidade segue-se que os conjuntos não têm ordem entre os seus elementos. Por isso, se queremos representar ordem, usamos *pares ordenados* $\langle x, y\rangle$. Num par não ordenado $\{x, y\}$, a ordem não importa: $\{x, y\} = \{y, x\}$. Num par ordenado, importa: se $x \neq y$, então $\langle x, y\rangle \neq \langle y, x\rangle$.

Como devemos pensar os pares ordenados na teoria dos conjuntos? O essencial é preservar a ideia de que pares ordenados são idênticos se e só se compartilham o mesmo primeiro elemento e o mesmo segundo elemento, isto é:

$$\langle a, b\rangle = \langle c, d\rangle \text{ se e só se } a = c \text{ e } b = d.$$

Podemos definir pares ordenados na teoria dos conjuntos usando a definição de Wiener–Kuratowski.

Definição 1.23 (Par ordenado). $\langle a,b\rangle = \{\{a\},\{a,b\}\}$.

Tendo fixado uma definição de par ordenado, podemos usá-la para definir outros conjuntos. Por exemplo, por vezes queremos sequências ordenadas de mais de dois objetos, p.ex., *triplos* $\langle x,y,z\rangle$, *quádruplos* $\langle x,y,z,u\rangle$, e assim por diante. Podemos pensar nos triplos como pares ordenados especiais, em que o primeiro elemento é ele próprio um par ordenado: $\langle x,y,z\rangle$ é $\langle\langle x,y\rangle,z\rangle$. O mesmo vale para quádruplos: $\langle x,y,z,u\rangle$ é $\langle\langle\langle x,y\rangle,z\rangle,u\rangle$, e assim por diante. Em geral, falamos de *n-tuplas ordenadas* $\langle x_1,\ldots,x_n\rangle$.

Certos conjuntos de pares ordenados, ou outras n-tuplas ordenadas, serão úteis.

Definição 1.24 (Produto cartesiano). Dados conjuntos A e B, o seu *produto cartesiano* $A\times B$ é definido por

$$A\times B = \{\langle x,y\rangle : x\in A \text{ e } y\in B\}.$$

Exemplo 1.25. Se $A=\{0,1\}$, e $B=\{1,a,b\}$, então o seu produto é

$$A\times B = \{\langle 0,1\rangle,\langle 0,a\rangle,\langle 0,b\rangle,\langle 1,1\rangle,\langle 1,a\rangle,\langle 1,b\rangle\}.$$

Exemplo 1.26. Se A é um conjunto, o produto de A consigo próprio, $A\times A$, também se escreve A^2. É o conjunto de *todos* os pares $\langle x,y\rangle$ com $x,y\in A$. O conjunto de todos os triplos $\langle x,y,z\rangle$ é A^3, e assim por diante. Podemos dar uma definição recursiva:

$$A^1 = A$$
$$A^{k+1} = A^k\times A$$

Proposição 1.27. *Se A tem n elementos e B tem m elementos, então $A \times B$ tem $n \cdot m$ elementos.*

Prova. Para todo elemento x em A, há m elementos da forma $\langle x, y\rangle \in A \times B$. Seja $B_x = \{\langle x, y\rangle : y \in B\}$. Como sempre que $x_1 \neq x_2$, $\langle x_1, y\rangle \neq \langle x_2, y\rangle$, tem-se $B_{x_1} \cap B_{x_2} = \emptyset$. Mas se $A = \{x_1, \ldots, x_n\}$, então $A \times B = B_{x_1} \cup \cdots \cup B_{x_n}$, e portanto tem $n \cdot m$ elementos.

Para visualizar isto, dispomos os elementos de $A \times B$ numa grade:

$$\begin{array}{rccccc} B_{x_1} = & \{\langle x_1, y_1\rangle & \langle x_1, y_2\rangle & \ldots & \langle x_1, y_m\rangle\} \\ B_{x_2} = & \{\langle x_2, y_1\rangle & \langle x_2, y_2\rangle & \ldots & \langle x_2, y_m\rangle\} \\ \vdots & & \vdots & & \\ B_{x_n} = & \{\langle x_n, y_1\rangle & \langle x_n, y_2\rangle & \ldots & \langle x_n, y_m\rangle\} \end{array}$$

Como os x_i são todos distintos, e os y_j são todos distintos, quaisquer dois pares nesta grade são distintos, e há $n \cdot m$ deles. □

Exemplo 1.28. Se A é um conjunto, uma *palavra* sobre A é qualquer sequência de elementos de A. Uma sequência pode ser vista como uma n-tupla de elementos de A. Por exemplo, se $A = \{a, b, c\}$, então a sequência "bac" pode ser vista como o triplo $\langle b, a, c\rangle$. Palavras, isto é, sequências de símbolos, são de importância crucial na ciência da computação. Por convenção, contamos os elementos de A como sequências de comprimento 1, e $\emptyset$ como a sequência de comprimento 0. O conjunto de *todas* as palavras sobre A é então

$$A^* = \{\emptyset\} \cup A \cup A^2 \cup A^3 \cup \ldots$$

1.6 Paradoxo de Russell

A extensionalidade autoriza a notação $\{x : \varphi(x)\}$, para *o* conjunto dos x tais que $\varphi(x)$. Contudo, tudo o que a extensionalidade *realmente* autoriza é o seguinte pensamento. *Se* há um conjunto cujos membros são exatamente os que satisfazem φ, *então* há apenas

um tal conjunto. Dito de outro modo: fixada alguma φ, o conjunto $\{x : \varphi(x)\}$ é único, *se existir*.

Mas este condicional é importante! Crucialmente, nem toda propriedade se presta à *compreensão*. Isto é, algumas propriedades *não* definem conjuntos. Se todas definissem, cairíamos em contradições flagrantes. O exemplo mais famoso disso é o Paradoxo de Russell.

Os conjuntos podem ser elementos de outros conjuntos—por exemplo, o conjunto potência de um conjunto A é constituído por conjuntos. Por isso faz sentido perguntar ou investigar se um conjunto é um elemento de outro conjunto. Pode um conjunto ser membro de si próprio? Nada na ideia de conjunto parece excluir isso. Por exemplo, se *todos* os conjuntos formam uma coleção de objetos, alguém poderia pensar que podem reunir-se num único conjunto—o conjunto de todos os conjuntos. E ele, sendo conjunto, seria um elemento do conjunto de todos os conjuntos.

O Paradoxo de Russell surge quando consideramos a propriedade de não ter a si próprio como um elemento, de ser *não auto-membro*. E se supusermos que há um conjunto de todos os conjuntos que não têm a si próprios como um elemento? Será que

$$R = \{x : x \notin x\}$$

existe? Verifica-se que podemos provar que não.

Teorema 1.29 (Paradoxo de Russell). *Não há conjunto* $R = \{x : x \notin x\}$.

Prova. Se $R = \{x : x \notin x\}$ existisse, então $R \in R$ se e só se $R \notin R$, o que é uma contradição. □

Vamos percorrer esta prova mais devagar. Se R existe, faz sentido perguntar se $R \in R$ ou não. Suponhamos que de fato $R \in R$. Agora, R foi definido como o conjunto de todos os conjuntos que não são elementos de si próprios. Logo, se $R \in R$,

então R não tem a propriedade definidora de R. Mas só os conjuntos que têm essa propriedade estão em R, logo R não pode ser um elemento de R, isto é, $R \notin R$. Mas R não pode ao mesmo tempo ser e não ser um elemento de R, logo temos uma contradição.

Como a suposição de que $R \in R$ leva a uma contradição, temos $R \notin R$. Mas isto também leva a uma contradição! Pois se $R \notin R$, então R tem a propriedade definidora de R, e assim R seria um elemento de R, tal como todos os outros conjuntos não auto-membros. E de novo, R não pode ao mesmo tempo não ser e ser um elemento de R.

Como montar uma teoria dos conjuntos que evite cair no Paradoxo de Russell, isto é, que evite afirmar de modo *inconsistente* que $R = \{x : x \notin x\}$ existe? Teríamos de estabelecer axiomas que nos dêem condições muito precisas para dizer quando os conjuntos existem (e quando não existem).

A teoria dos conjuntos esboçada neste capítulo não faz isso. É *genuinamente ingênua.* Diz apenas que os conjuntos obedecem à extensionalidade e que, se temos alguns conjuntos, podemos formar a sua união, interseção, etc. É possível desenvolver a teoria dos conjuntos com mais rigor do que isto.

Resumo

Um **conjunto** é uma coleção de objetos, os elementos do conjunto. Escrevemos $x \in A$ se x é um elemento de A. Conjuntos são **extensionais**—eles são completamente determinados por seus elementos. Conjuntos são especificados **listando** os elementos explicitamente ou fornecendo uma propriedade que os elementos compartilham (**abstração**). Extensionalidade significa que a ordem ou a maneira de listar ou especificar os elementos de um conjunto não importa. Para provar que A e B são o mesmo conjunto ($A = B$), deve-se provar que todo elemento de X é um elemento de Y e vice-versa.

Conjuntos importantes incluem os números naturais ($\mathbb{N}$), inteiros ($\mathbb{Z}$), racionais ($\mathbb{Q}$) e reais ($\mathbb{R}$), mas também **cadeias** (X^*) e **sequências** infinitas (X^ω) de objetos. A é um **subconjunto** de B, $A \subseteq B$, se todo elemento de A é também um elemento de B. A coleção de todos os subconjuntos de um conjunto B é, ela própria, um conjunto, o **conjunto potência** $\wp(B)$ de B. Podemos formar a **união** $A \cup B$ e a **interseção** $A \cap B$ de conjuntos. Um **par ordenado** $\langle x, y\rangle$ consiste em dois objetos x e y, mas naquela ordem específica. Os pares $\langle x, y\rangle$ e $\langle y, x\rangle$ são pares diferentes (a menos que $x = y$). O conjunto de todos os pares $\langle x, y\rangle$ em que $x \in A$ e $y \in B$ é chamado de **produto cartesiano** $A \times B$ de A e B. Escrevemos A^2 para $A \times A$; assim, por exemplo, $\mathbb{N}^2$ é o conjunto dos pares de números naturais.

Problemas

Problema 1.1. Prove que há no máximo um conjunto vazio, isto é, mostre que se A e B são conjuntos sem elementos, então $A = B$.

Problema 1.2. Liste todos os subconjuntos de $\{a, b, c, d\}$.

Problema 1.3. Mostre que se A tem n elementos, então $\wp(A)$ tem 2^n elementos.

Problema 1.4. Prove que se $A \subseteq B$, então $A \cup B = B$.

Problema 1.5. Prove rigorosamente que se $A \subseteq B$, então $A \cap B = A$.

Problema 1.6. Mostre que se A é um conjunto e $A \in B$, então $A \subseteq \bigcup B$.

Problema 1.7. Prove que se $A \subsetneq B$, então $B \setminus A \neq \emptyset$.

Problema 1.8. Usando Definição 1.23, prove que $\langle a, b\rangle = \langle c, d\rangle$ se e só se $a = c$ e $b = d$.

Problema 1.9. Liste todos os elementos de $\{1,2,3\}^3$.

Problema 1.10. Mostre, por indução em k, que para todo $k \geq 1$, se A tem n elementos, então A^k tem n^k elementos.

CAPÍTULO 2

Relações

2.1 Relações como Conjuntos

Na seção 1.3, mencionamos alguns conjuntos importantes: $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$. Sem dúvida que se lembram de algumas relações interessantes entre os elementos de alguns destes conjuntos. Por exemplo, cada um destes conjuntos tem uma *relação de ordem* completamente padrão sobre si. Há também a relação *ser idêntico a*, segundo a qual cada objeto só se relaciona consigo próprio e com nenhum outro. Há muitas outras relações interessantes que encontraremos, e ainda mais relações possíveis. Antes de as revermos, contudo, começaremos por observar que podemos ver relações como um tipo especial de conjunto.

Para isso, recordemos duas coisas da seção 1.5. Primeiro, recordemos a noção de *par ordenado*: dados a e b, podemos formar $\langle a, b\rangle$. É importante notar que a ordem dos elementos *importa* aqui. Assim, se $a \neq b$ então $\langle a, b\rangle \neq \langle b, a\rangle$. (Contrastemos isto com pares não ordenados, isto é, conjuntos de 2 elementos, onde $\{a, b\} = \{b, a\}$.) Segundo, recordemos a noção de *produto cartesiano*: se A e B são conjuntos, então podemos formar $A \times B$, o conjunto de todos os pares $\langle x, y\rangle$ com $x \in A$ e $y \in B$. Em particular, $A^2 = A \times A$ é o conjunto de todos os pares ordenados de A.

Consideremos agora uma relação particular num conjunto: a relação $<$ no conjunto $\mathbb{N}$ dos números naturais. Seja o conjunto

de todos os pares de números $\langle n, m\rangle$ onde $n < m$, isto é,

$$R = \{\langle n, m\rangle : n, m \in \mathbb{N} \text{ e } n < m\}.$$

Há uma ligação estreita entre n ser menor que m, e o par $\langle n, m\rangle$ ser membro de R, nomeadamente:

$$n < m \text{ se e só se } \langle n, m\rangle \in R.$$

De fato, sem qualquer perda de informação, podemos considerar o conjunto R como *sendo* a relação $<$ em $\mathbb{N}$.

Do mesmo modo podemos construir um subconjunto de $\mathbb{N}^2$ para qualquer relação entre números. Reciprocamente, dado qualquer conjunto de pares de números $S \subseteq \mathbb{N}^2$, há uma relação correspondente entre números, nomeadamente, a relação segundo a qual n se liga a m se e só se $\langle n, m\rangle \in S$. Isto justifica a definição seguinte:

Definição 2.1 (Relação binária). Uma *relação binária* num conjunto A é um subconjunto de A^2. Se $R \subseteq A^2$ é uma relação binária em A e $x, y \in A$, por vezes escrevemos Rxy (ou xRy) para $\langle x, y\rangle \in R$.

Exemplo 2.2. O conjunto $\mathbb{N}^2$ dos pares de números naturais pode ser listado numa matriz bidimensional assim:

$$\begin{array}{ccccc}
\langle \mathbf{0},\mathbf{0}\rangle & \langle 0,1\rangle & \langle 0,2\rangle & \langle 0,3\rangle & \dots \\
\langle 1,0\rangle & \langle \mathbf{1},\mathbf{1}\rangle & \langle 1,2\rangle & \langle 1,3\rangle & \dots \\
\langle 2,0\rangle & \langle 2,1\rangle & \langle \mathbf{2},\mathbf{2}\rangle & \langle 2,3\rangle & \dots \\
\langle 3,0\rangle & \langle 3,1\rangle & \langle 3,2\rangle & \langle \mathbf{3},\mathbf{3}\rangle & \dots \\
\vdots & \vdots & \vdots & \vdots & \ddots
\end{array}$$

Colocamos a diagonal, aqui, em negrito, pois o subconjunto de $\mathbb{N}^2$ constituído pelos pares que estão na diagonal, isto é,

$$\{\langle 0,0\rangle, \langle 1,1\rangle, \langle 2,2\rangle, \dots\},$$

é a *relação identidade em* $\mathbb{N}$. (Como a relação identidade é popular, definamos $\mathrm{Id}_A = \{\langle x,x\rangle : x \in A\}$ para qualquer conjunto A.) O subconjunto de todos os pares acima da diagonal, isto é,

$$L = \{\langle 0,1\rangle, \langle 0,2\rangle, \ldots, \langle 1,2\rangle, \langle 1,3\rangle, \ldots, \langle 2,3\rangle, \langle 2,4\rangle, \ldots\},$$

é a relação *menor que*, isto é, Lnm se e só se $n < m$. O subconjunto dos pares abaixo da diagonal, isto é,

$$G = \{\langle 1,0\rangle, \langle 2,0\rangle, \langle 2,1\rangle, \langle 3,0\rangle, \langle 3,1\rangle, \langle 3,2\rangle, \ldots\},$$

é a relação *maior que*, isto é, Gnm se e só se $n > m$. A união de L com I, que podemos chamar $K = L \cup I$, é a relação *menor ou igual a*: Knm se e só se $n \leq m$. Do mesmo modo, $H = G \cup I$ é a *relação maior ou igual a.* Estas relações L, G, K, e H são tipos especiais de relações chamadas *ordens.* L e G gozam da propriedade de que nenhum número está em relação L ou G consigo próprio (isto é, para todo n, não se verifica nem Lnn nem Gnn). Relações com esta propriedade chamam-se *irreflexivas*, e, se também forem ordens, chamam-se *ordens estritas.*

Embora ordens e identidade sejam relações importantes e naturais, convém enfatizar que, segundo a nossa definição, *qualquer* subconjunto de A^2 é uma relação em A, por mais artificial ou forçado que pareça. Em particular, $\emptyset$ é uma relação em qualquer conjunto (a *relação vazia*, que nenhum par de elementos satisfaz), e A^2 em si é também uma relação em A (uma que todo par de elementos satisfaz), chamada a *relação universal.* Mas também algo como $E = \{\langle n,m\rangle : n > 5 \text{ ou } m \times n \geq 34\}$ conta como relação.

2.2 Propriedades Especiais das Relações

Alguns tipos de relações revelam-se tão comuns que receberam nomes especiais. Por exemplo, $\leq$ e $\subseteq$ relacionam os seus respectivos domínios (digamos, $\mathbb{N}$ no caso de $\leq$ e $\wp(A)$ no caso de $\subseteq$) de modos semelhantes. Para precisar exatamente como estas relações são semelhantes, e como diferem, as categorizamos

segundo algumas propriedades especiais que as relações podem ter. Verifica-se que (combinações de) algumas destas propriedades especiais são especialmente importantes: ordens e relações de equivalência.

Definição 2.3 (Reflexividade). Uma relação $R \subseteq A^2$ é *reflexiva* se e só se, para todo $x \in A$, Rxx.

Definição 2.4 (Transitividade). Uma relação $R \subseteq A^2$ é *transitiva* se e só se, sempre que Rxy e Ryz, então também Rxz.

Definição 2.5 (Simetria). Uma relação $R \subseteq A^2$ é *simétrica* se e só se, sempre que Rxy, então também Ryx.

Definição 2.6 (Antissimetria). Uma relação $R \subseteq A^2$ é *antissimétrica* se e só se, sempre que tanto Rxy quanto Ryx, então $x = y$ (ou, noutras palavras: se $x \neq y$ então ou $\neg Rxy$ ou $\neg Ryx$).

Numa relação simétrica, Rxy e Ryx verificam-se sempre em conjunto, ou nenhuma se verifica. Numa relação antissimétrica, a única forma de Rxy e Ryx se verificarem em conjunto é que $x = y$. Note-se que isto não *exige* que Rxy e Ryx se verifiquem quando $x = y$, apenas que isso não está excluído. Assim, uma relação antissimétrica pode ser reflexiva, mas não é o caso que toda relação antissimétrica seja reflexiva. Note-se também que ser antissimétrica e ser meramente não simétrica são condições diferentes. De fato, uma relação pode ser ao mesmo tempo simétrica e antissimétrica (por exemplo, a relação identidade o é).

Definição 2.7 (Conexidade). Uma relação $R \subseteq A^2$ é *conexa* se para todos $x, y \in A$, se $x \neq y$, então ou Rxy ou Ryx.

Definição 2.8 (Irreflexividade). Uma relação $R \subseteq A^2$ diz-se *irreflexiva* se, para todo $x \in A$, não Rxx.

Definição 2.9 (Assimetria). Uma relação $R \subseteq A^2$ diz-se *assimétrica* se para nenhum par $x, y \in A$ se tem tanto Rxy quanto Ryx.

Note-se que se $A \neq \emptyset$, então nenhuma relação irreflexiva em A é reflexiva e toda relação assimétrica em A é também antissimétrica. Contudo, há $R \subseteq A^2$ que não são reflexivas nem irreflexivas, e há relações antissimétricas que não são assimétricas.

2.3 Relações de Equivalência

A relação identidade num conjunto é reflexiva, simétrica e transitiva. Relações R que têm todas as três destas propriedades são muito comuns.

Definição 2.10 (Relação de equivalência). Uma relação $R \subseteq A^2$ que é reflexiva, simétrica e transitiva chama-se uma *relação de equivalência.* Elementos x e y de A dizem-se *R-equivalentes* se Rxy.

As relações de equivalência dão origem à noção de *classe de equivalência.* Uma relação de equivalência "reparte" o domínio em partições distintas. Dentro de cada partição, todos os objetos estão relacionados uns com os outros; e objetos de partições diferentes não se relacionam entre si. Por vezes, é útil falar destas partições *diretamente.* Para esse fim, introduzimos uma definição:

Definição 2.11. Seja $R \subseteq A^2$ uma relação de equivalência. Para cada $x \in A$, a *classe de equivalência* de x em A é o conjunto $[x]_R = \{y \in A : Rxy\}$. O *quociente* de A por R é $A/_R = \{[x]_R : x \in A\}$, isto é, o conjunto destas classes de equivalência.

O resultado seguinte justifica a definição de classe de equivalência, provando que as classes de equivalência são de fato as partições de A:

Proposição 2.12. *Se $R \subseteq A^2$ é uma relação de equivalência, então Rxy se e só se $[x]_R = [y]_R$.*

Prova. Para o sentido da esquerda para a direita, suponhamos Rxy, e seja $z \in [x]_R$. Por definição, então, Rxz. Como R é uma relação de equivalência, Ryz. (Explicando melhor: como Rxy e R é simétrica temos Ryx, e como Rxz e R é transitiva temos Ryz.) Logo $z \in [y]_R$. Generalizando, $[x]_R \subseteq [y]_R$. Mas de modo perfeitamente análogo, $[y]_R \subseteq [x]_R$. Logo $[x]_R = [y]_R$, pela extensionalidade.

Para o sentido da direita para a esquerda, suponhamos $[x]_R = [y]_R$. Como R é reflexiva, Ryy, logo $y \in [y]_R$. Assim também $y \in [x]_R$ pela suposição de que $[x]_R = [y]_R$. Logo Rxy. □

Exemplo 2.13. Um bom exemplo de relações de equivalência vem da aritmética modular. Para quaisquer a, b, e $n \in \mathbb{Z}^+$, digamos que $a \equiv_n b$ se e só se dividir a por n dá o mesmo resto que dividir b por n. (De modo um pouco mais simbólico: $a \equiv_n b$ se e só se, para algum $k \in \mathbb{Z}$, $a - b = kn$.) Agora, $\equiv_n$ é uma relação de equivalência, para qualquer n. E há exatamente n classes de equivalência distintas geradas por $\equiv_n$; isto é, $\mathbb{N}/_{\equiv_n}$ tem n elementos. São: o conjunto dos números divisíveis por n sem resto, isto é, $[0]_{\equiv_n}$; o conjunto dos números divisíveis por n com resto 1, isto é, $[1]_{\equiv_n}$; ...; e o conjunto dos números divisíveis por n com resto $n - 1$, isto é, $[n-1]_{\equiv_n}$.

2.4 Ordens

Muitas das nossas comparações envolvem descrever certos objetos como sendo "menores que", "iguais a" ou "maiores que" outros objetos, sob algum aspecto. Isso envolve relações de *ordem*. Mas há vários tipos de relação de ordem. Por exemplo, algumas exigem que quaisquer dois objetos sejam comparáveis, outras não. Algumas incluem a identidade (como $\leq$) e outras a excluem (como $<$). Ter aqui uma taxonomia ajuda-nos.

Definição 2.14 (Pré-ordem). Uma relação que é ao mesmo tempo reflexiva e transitiva chama-se uma *pré-ordem*.

Definição 2.15 (Ordem parcial). Uma pré-ordem que também é antissimétrica chama-se uma *ordem parcial*.

Definição 2.16 (Ordem linear). Uma ordem parcial que também é conexa chama-se uma *ordem total* ou *ordem linear*.

Exemplo 2.17. Toda ordem linear é também uma ordem parcial, e toda ordem parcial é também uma pré-ordem, mas as implicações recíprocas não valem. A relação universal sobre A é uma pré-ordem, pois é reflexiva e transitiva. Mas, se A tem mais de um elemento, a relação universal não é antissimétrica e, portanto, não é uma ordem parcial.

Exemplo 2.18. Considere a relação *não mais longa que* $\preccurlyeq$ sobre $\mathbb{B}^*$: $x \preccurlyeq y$ sse $\text{len}(x) \leq \text{len}(y)$. Trata-se de uma pré-ordem (reflexiva e transitiva), e até conexa, mas não é uma ordem parcial, pois não é antissimétrica. Por exemplo, $01 \preccurlyeq 10$ e $10 \preccurlyeq 01$, mas $01 \neq 10$.

Exemplo 2.19. Uma ordem parcial importante é a relação $\subseteq$ sobre um conjunto de conjuntos. Em geral não é uma ordem linear:

se $a \neq b$ e consideramos $\wp(\{a,b\}) = \{\emptyset, \{a\}, \{b\}, \{a,b\}\}$, vemos que $\{a\} \not\subseteq \{b\}$ e $\{a\} \neq \{b\}$ e $\{b\} \not\subseteq \{a\}$.

Exemplo 2.20. A relação de *divisibilidade sem resto* nos dá uma ordem parcial que não é linear. Para inteiros n e m, escrevemos $n \mid m$ para significar que n divide m (exatamente), isto é, sse existe algum inteiro k tal que $m = kn$. Sobre $\mathbb{N}$, isso é uma ordem parcial, mas não linear: por exemplo, $2 \nmid 3$ e também $3 \nmid 2$. Vista como relação sobre $\mathbb{Z}$, a divisibilidade é apenas uma pré-ordem, pois não é antissimétrica: $1 \mid -1$ e $-1 \mid 1$ mas $1 \neq -1$.

Exemplo 2.21. A relação de *extensão* sobre um conjunto de sequências A^* é a seguinte: $s \sqsubseteq s'$ sse $s = \Lambda$ (a sequência vazia), $s = s'$, ou $s = \langle s_1, \ldots, s_n \rangle$ e $s' = \langle s_1, \ldots, s_n, s_{n+1}, \ldots, s_m \rangle$. Se $s \sqsubseteq s'$ dizemos também que s é um *segmento inicial* de s'. A relação de extensão sobre A^* é uma ordem parcial, mas não linear, p.ex., se $a \neq b$, então $ab \not\sqsubseteq ba$ e $ba \not\sqsubseteq ab$.

Definição 2.22 (Ordem estrita). Uma *ordem estrita* é uma relação irreflexiva, assimétrica e transitiva.

Definição 2.23 (Ordem linear estrita). Uma ordem estrita que também é conexa chama-se *ordem total estrita* ou *ordem linear estrita.*

Exemplo 2.24. $\leq$ é a ordem linear correspondente à ordem linear estrita $<$. $\subseteq$ é a ordem parcial correspondente à ordem estrita $\subsetneq$.

Qualquer ordem estrita R sobre A pode transformar-se numa ordem parcial adicionando a diagonal Id_A, isto é, adicionando todos os pares $\langle x, x \rangle$. (Isto chama-se o *fecho reflexivo* de R.) Reciprocamente, a partir de uma ordem parcial obtém-se uma ordem estrita removendo Id_A. Os dois resultados seguintes tornam isto preciso.

Proposição 2.25. *Se R é uma ordem estrita sobre A, então $R^+ = R \cup \mathrm{Id}_A$ é uma ordem parcial. Além disso, se R é uma ordem linear estrita, então R^+ é uma ordem linear.*

Prova. Suponhamos que R é uma ordem estrita, isto é, $R \subseteq A^2$ e R é irreflexiva, assimétrica e transitiva. Seja $R^+ = R \cup \mathrm{Id}_A$. Temos de mostrar que R^+ é reflexiva, antissimétrica e transitiva.

R^+ é claramente reflexiva, pois $\langle x, x\rangle \in \mathrm{Id}_A \subseteq R^+$ para todo $x \in A$.

Para mostrar que R^+ é antissimétrica, suponhamos, para redução ao absurdo, que R^+xy e R^+yx mas $x \neq y$. Como $\langle x, y\rangle \in R \cup \mathrm{Id}_A$, mas $\langle x, y\rangle \notin \mathrm{Id}_A$, temos de ter $\langle x, y\rangle \in R$, isto é, Rxy. Do mesmo modo, Ryx. Mas isto contradiz a hipótese de que R é assimétrica.

Para estabelecer a transitividade, suponhamos que R^+xy e R^+yz. Se tanto $\langle x, y\rangle \in R$ como $\langle y, z\rangle \in R$, então $\langle x, z\rangle \in R$, pois R é transitiva. Caso contrário, ou $\langle x, y\rangle \in \mathrm{Id}_A$, isto é, $x = y$, ou $\langle y, z\rangle \in \mathrm{Id}_A$, isto é, $y = z$. No primeiro caso, temos R^+yz por hipótese, $x = y$, logo R^+xz. Analogamente no segundo caso. Em qualquer caso, R^+xz; assim, R^+ é também transitiva.

Quanto à cláusula "além disso", suponhamos que R é também conexa. Então, para todo $x \neq y$, ou Rxy ou Ryx, isto é, ou $\langle x, y\rangle \in R$ ou $\langle y, x\rangle \in R$. Como $R \subseteq R^+$, isto continua a valer para R^+, logo R^+ é também conexa. □

Proposição 2.26. *Se R é uma ordem parcial sobre A, então $R^- = R \setminus \mathrm{Id}_A$ é uma ordem estrita. Além disso, se R é uma ordem linear, então R^- é uma ordem linear estrita.*

Prova. Deixamos como exercício. □

O resultado simples a seguir estabelece que ordens lineares estritas satisfazem uma propriedade no estilo da extensionalidade:

Proposição 2.27. *Se $<$ é uma ordem linear estrita sobre A, então:*

$$(\forall a, b \in A)((\forall x \in A)(x < a \leftrightarrow x < b) \rightarrow a = b).$$

Prova. Suponhamos $(\forall x \in A)(x < a \leftrightarrow x < b)$. Se $a < b$, então $a < a$, o que contradiz o fato de $<$ ser irreflexiva; logo $a \not< b$. De forma totalmente análoga, $b \not< a$. Logo $a = b$, pois $<$ é conexa.□

2.5 Grafos

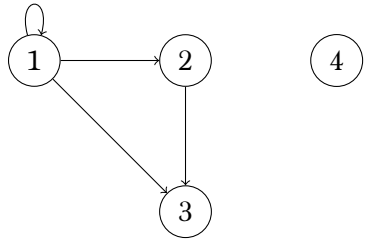
Um *grafo* é um diagrama no qual pontos—chamados "nós" ou "vértices" (plural de "vértice")—são conectados por arestas. Grafos são uma ferramenta onipresente em matemática discreta e em ciência da computação. São incrivelmente úteis para representar e visualizar relações e estruturas, desde coisas concretas como redes de vários tipos até estruturas abstratas como os possíveis resultados de decisões. Existem muitos tipos diferentes de grafos na literatura que diferem, p.ex., quanto a se as arestas são direcionadas ou não, têm rótulos ou não, se pode haver arestas de um nó para o mesmo nó, múltiplas arestas entre os mesmos nós, etc. *Grafos direcionados* têm uma conexão especial com relações.

Definição 2.28 (Grafo direcionado). Um *grafo direcionado* $G = \langle V, E \rangle$ é um conjunto de *vértices* V e um conjunto de *arestas* $E \subseteq V^2$.

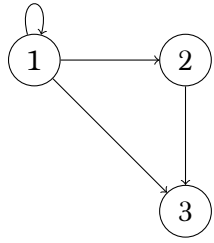
De acordo com nossa definição, um grafo é simplesmente um conjunto junto com uma relação nesse conjunto. Claro, ao falar de grafos, é natural esperar que sejam representados graficamente: podemos desenhar um grafo conectando dois vértices v_1 e v_2 por uma seta sse $\langle v_1, v_2 \rangle \in E$. A única diferença entre uma relação por si só e um grafo é que um grafo especifica o conjunto de vértices, isto é, um grafo pode ter vértices isolados. O ponto importante, porém, é que toda relação R em um conjunto X pode ser vista como um grafo direcionado $\langle X, R \rangle$, e reciprocamente,

um grafo direcionado $\langle V, E\rangle$ pode ser visto como uma relação $E \subseteq V^2$ com o conjunto V explicitamente especificado.

Exemplo 2.29. O grafo $\langle V, E\rangle$ com $V = \{1,2,3,4\}$ e $E = \{\langle 1,1\rangle, \langle 1,2\rangle, \langle 1,3\rangle, \langle 2,3\rangle\}$ se parece com isto:



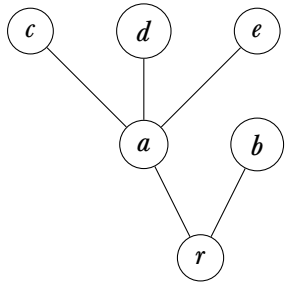
Este é um grafo diferente de $\langle V', E\rangle$ com $V' = \{1,2,3\}$, que se parece com isto:



2.6 Árvores

Um tipo particular de ordem parcial que desempenha um papel importante em todas as partes da lógica é uma *árvore*. Árvores finitas aparecem em partes elementares da lógica: por exemplo, fórmulas podem ser entendidas em termos de sua decomposição em uma árvore sintática, enquanto derivações em muitos sistemas de derivação também tomam a forma de árvores finitas. Árvores infinitas aparecem já na prova dos teoremas de completude para a lógica proposicional e de primeira ordem, e são usadas em toda a lógica matemática.

O conceito de árvore da teoria dos conjuntos está intimamente relacionado à noção de árvore na teoria dos grafos. Aqui está uma figura de uma árvore (finita):



O nó mais baixo r é a raiz. Todo nó diferente de r tem exatamente um nó pai imediatamente abaixo dele. Podemos pensar na relação em que um nó x está para um nó y se y pode ser alcançado a partir de x seguindo arestas para cima como x sendo um *ancestral* de y.

A relação de ancestralidade em uma árvore é uma ordem parcial estrita. Isso motiva a definição teórico-conjuntista. Para enunciá-la, precisamos de dois conceitos. Um *menor elemento* em um conjunto A parcialmente ordenado por $\leq$ é um elemento $x \in A$ tal que para todo $y \in A$ temos que $x \leq y$. Um conjunto é *bem ordenado* por $\leq$ se cada um de seus subconjuntos não vazios tem um menor elemento.

Definição 2.30 (Árvore). Uma *árvore* é um par $T = \langle A, \leq \rangle$ tal que A é um conjunto e $\leq$ é uma ordem parcial em A com um único menor elemento $r \in A$ (chamado de *raiz*) tal que para todo $x \in A$, o conjunto $\{y : y \leq x\}$ é bem ordenado por $\leq$.

Definição 2.31 (Sucessores). Suponha $T = \langle A, \leq \rangle$ é uma árvore. Se $x, y \in A$, $x < y$, e não existe $z \in A$ tal que $x < z < y$, então dizemos que y é um *sucessor* de x.

Os sucessores de $x \in A$ também são chamados de seus *filhos*. Se y é um sucessor de x, então chamamos x de *predecessor* ou *pai* de y.

Proposição 2.32. *Se $\langle A, \leq \rangle$ é uma árvore, então todo $x \in A$ diferente da raiz tem no máximo um predecessor.*

Prova. Suponha $y_1 < x$ e $y_2 < x$ e $y_1 \neq y_2$. Então $\{y_1, y_2\} \subseteq \{z : z < x\}$. Como $\{z : z < x\}$ é bem ordenado por $\leq$, seu subconjunto $\{y_1, y_2\}$ tem um menor elemento, que obviamente deve ser ou y_1 ou y_2. Então ou $y_1 \leq y_2$ ou $y_2 \leq y_1$. Supusemos que $y_1 \neq y_2$, então na verdade ou $y_1 < y_2$ ou $y_2 < y_1$. Como supusemos que $y_1 < x$ e $y_2 < x$, temos ainda que ou $y_1 < y_2 < x$ ou $y_2 < y_1 < x$. Então y_1 e y_2 não podem ambos ser predecessores de x. □

Definição 2.33. Uma árvore $T = \langle A, \leq \rangle$ é dita *infinita* se A é um conjunto infinito, e *finita* caso contrário. Se T é tal que todo $x \in A$ tem apenas um número finito de sucessores, então dizemos que T é *finitamente ramificada.*

Definição 2.34 (Ramos). Dada uma árvore $T = \langle A, \leq \rangle$, um *ramo* de T é uma cadeia maximal em T, isto é, um conjunto $B \subseteq A$ tal que para quaisquer $x, y \in B$ ou $x \leq y$ ou $y \leq x$, e para qualquer $z \in X \setminus B$ existe $u \in B$ tal que nem $z \leq u$ nem $u \leq z$. Usamos $[T]$ para denotar o conjunto de todos os ramos de T.

Exemplo 2.35. Um exemplo clássico de árvore finitamente ramificada é a *árvore binária infinita* de sequências finitas de 0s e 1s, às vezes denotada $\{0,1\}^*$ ou $\mathbb{B}^*$, ordenada pela relação de extensão $\sqsubseteq$ (p.ex., $101 \sqsubseteq 101101$). Como qualquer cadeia binária pode sempre ser estendida adicionando um 0 ou um 1 no final, esta árvore contém infinitos elementos: todo elemento s tem exatamente dois sucessores, $s0$ e $s1$. Sua raiz é a sequência vazia Λ.

Exemplo 2.36. De maneira um pouco mais geral, o conjunto de sequências finitas de números naturais $\mathbb{N}^*$ com a relação de extensão $\sqsubseteq$ também é uma árvore. É obviamente não finitamente ramificada: todo $s \in \mathbb{N}^*$ tem infinitos sucessores sn, um para cada $n \in \mathbb{N}$. Todo $A \subseteq \mathbb{N}^*$ que é fechado sob $\sqsubseteq$ é uma *subárvore* de $\mathbb{N}^*$. (Isto é, A é tal que se $s \in A$ e $s' \sqsubseteq s$, então também $s' \in A$.) Todas as árvores finitas podem ser representadas como subárvores finitas de $\mathbb{N}^*$.

Proposição 2.37 (Lema de Kőnig). *Se $T = \langle A, \leq \rangle$ é uma árvore infinita finitamente ramificada, então T tem um ramo infinito.*

Um caso especial do Lema de Kőnig amplamente usado em teoria da computabilidade, conhecido como *Lema Fraco de Kőnig*, é o seguinte: qualquer subárvore infinita de $\{0,1\}^*$ tem um ramo infinito.

2.7 Operações sobre Relações

É frequentemente útil modificar ou combinar relações. Na Proposição 2.25, consideramos a *união* de relações, que é apenas a união de duas relações vistas como conjuntos de pares. Do mesmo modo, na Proposição 2.26, consideramos a diferença relativa de relações. Eis algumas outras operações que podemos efetuar sobre relações.

Definição 2.38. Sejam R e S relações, e A um conjunto qualquer.

A *inversa* de R é $R^{-1} = \{\langle y, x\rangle : \langle x, y\rangle \in R\}$.
O *produto relativo* de R e S é $(R \mid S) = \{\langle x, z\rangle : \exists y(Rxy \wedge Syz)\}$.
A *restrição* de R a A é $R\restriction_A = R \cap A^2$.
A *aplicação* de R a A é $R[A] = \{y : (\exists x \in A)Rxy\}$

Exemplo 2.39. Seja $S \subseteq \mathbb{Z}^2$ a relação sucessora sobre $\mathbb{Z}$, isto é, $S = \{\langle x, y\rangle \in \mathbb{Z}^2 : x + 1 = y\}$, de modo que Sxy sse $x + 1 = y$.

S^{-1} é a relação antecessora sobre $\mathbb{Z}$, isto é, $\{\langle x,y\rangle \in \mathbb{Z}^2 : x-1=y\}$.

$S \mid S$ é $\{\langle x,y\rangle \in \mathbb{Z}^2 : x+2=y\}$

$S\restriction_{\mathbb{N}}$ é a relação sucessora sobre $\mathbb{N}$.

$S[\{1,2,3\}]$ é $\{2,3,4\}$.

Definição 2.40 (Fecho transitivo). Seja $R \subseteq A^2$ uma relação binária.

O *fecho transitivo* de R é $R^+ = \bigcup_{0<n\in\mathbb{N}} R^n$, onde definimos recursivamente $R^1 = R$ e $R^{n+1} = R^n \mid R$.

O *fecho reflexivo transitivo* de R é $R^* = R^+ \cup \mathrm{Id}_A$.

Exemplo 2.41. Tome a relação sucessora $S \subseteq \mathbb{Z}^2$. S^2xy sse $x+2=y$, S^3xy sse $x+3=y$, etc. Logo S^+xy sse $x+n=y$ para algum $n \geq 1$. Em outras palavras, S^+xy sse $x<y$, e S^*xy sse $x \leq y$.

Resumo

Uma **relação** R sobre um conjunto A é um modo de relacionar elementos de A. Escrevemos Rxy se a relação **vale** entre x e y. Formalmente, podemos considerar R como o conjunto de pares $\langle x,y\rangle \in A^2$ tais que Rxy. Ser menor que, maior que, igual a, dividir exatamente, ter o mesmo comprimento que, ser um subconjunto de, e ter o mesmo tamanho que são todos exemplos importantes de relações (sobre conjuntos de números, cadeias, ou de conjuntos). **Grafos** são um modo geral de representar relações visualmente. Mas um grafo também pode ser visto como uma relação binária (a relação **aresta**) junto com o conjunto subjacente de **vértices**.

Algumas relações compartilham certas características que as tornam especialmente interessantes ou úteis. Uma relação R é **reflexiva** se tudo está R-relacionado consigo mesmo; **simétrica**, se, com Rxy, também Ryx vale para quaisquer x e y; e **transitiva** se Rxy e Ryz garantem Rxz. As relações que têm todas essas três

propriedades são **relações de equivalência**. Uma relação é **anti-simétrica** se Rxy e Ryx garantem $x = y$. **Ordens parciais** são aquelas relações que são reflexivas, anti-simétricas e transitivas. Uma **ordem linear** é qualquer ordem parcial que satisfaz que, para quaisquer x e y, ou Rxy ou $x = y$ ou Ryx. (Em geral, uma relação com essa propriedade é **conexa**).

Como as relações são conjuntos (de pares), elas podem ser operadas como conjuntos (por exemplo, podemos formar a união e a interseção de relações). Também podemos encadeá-las (**produto relativo** $R \mid S$). Se formarmos o produto relativo de R consigo mesmo um número arbitrário de vezes, obtemos o **fecho transitivo** R^+ de R.

Problemas

Problema 2.1. Liste os elementos da relação $\subseteq$ no conjunto $\wp(\{a, b, c\})$.

Problema 2.2. Dê exemplos de relações que sejam (a) reflexivas e simétricas mas não transitivas, (b) reflexivas e antissimétricas, (c) antissimétricas, transitivas, mas não reflexivas, e (d) reflexivas, simétricas e transitivas. Não use relações sobre números ou conjuntos.

Problema 2.3. Mostre que $\equiv_n$ é uma relação de equivalência, para qualquer $n \in \mathbb{Z}^+$, e que $\mathbb{N}/_{\equiv_n}$ tem exatamente n membros.

Problema 2.4. Apresente uma prova da Proposição 2.26.

Problema 2.5. Considere a relação menor ou igual a $\leq$ no conjunto $\{1, 2, 3, 4\}$ como um grafo e desenhe o diagrama correspondente.

Problema 2.6. Mostre que o fecho transitivo de R é de fato transitivo.

CAPÍTULO 3

Funções

3.1 Noções Básicas

Uma *função* é um mapa que envia cada elemento de um dado conjunto para um elemento específico em algum (outro) conjunto dado. Por exemplo, a operação de somar 1 define uma função: cada número n é mapeado para um único número $n+1$.

Mais em geral, funções podem tomar pares, triplos, etc., como entradas e devolver algum tipo de saída. Muitas funções nos são familiares da aritmética básica. Por exemplo, a adição e a multiplicação são funções. Tomam dois números e devolvem um terceiro.

Neste sentido matemático e abstrato, uma função é uma *caixa preta*: só importa que saída fica associada a que entrada, não o método de calcular a saída.

Definição 3.1 (Função). Uma *função* $f : A \to B$ é um mapeamento de cada elemento de A para um elemento de B.

Chamamos A de *domínio* de f e B de *contradomínio* de f. Os elementos de A chamam-se entradas ou *argumentos* de f, e o elemento de B que fica associado a um argumento x por f chama-se o *valor de* f para o argumento x, escrevendo-se $f(x)$.

A *imagem* $\mathrm{Im}(f)$ de f é o subconjunto do contradomínio constituído pelos valores de f para algum argumento; $\mathrm{Im}(f) = \{f(x) : x \in A\}$.

O diagrama na Figura 3.1 pode ajudar a pensar em funções. A elipse à esquerda representa o *domínio* da função; a elipse à direita representa o *contradomínio*; e uma seta aponta de um *argumento* no domínio para o *valor* correspondente no contradomínio.

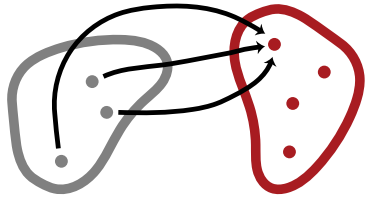


Figura 3.1: Uma função é um mapeamento de cada elemento de um conjunto para um elemento de outro. Uma seta vai de um argumento no domínio para o valor correspondente no contradomínio.

Exemplo 3.2. A multiplicação toma pares de números naturais como entradas e mapeia-os para números naturais como saídas, logo vai de $\mathbb{N} \times \mathbb{N}$ (o domínio) para $\mathbb{N}$ (o contradomínio). Como se vê, a imagem é também $\mathbb{N}$, pois todo $n \in \mathbb{N}$ é $n \times 1$.

Exemplo 3.3. A multiplicação é uma função porque associa cada entrada—cada par de números naturais—a uma única saída: $\times\colon \mathbb{N}^2 \to \mathbb{N}$. Em contraste, mapear um número natural n para um real x tal que $x^2 = n$ não é funcional, pois cada inteiro positivo n tem duas raízes quadradas: $\sqrt{n}$ e $-\sqrt{n}$. Podemos torná-la funcional devolvendo só a raiz quadrada positiva: $\sqrt{\ } : \mathbb{N} \to \mathbb{R}$.

Exemplo 3.4. A relação que associa cada aluno de uma turma à sua nota final é uma função—nenhum aluno pode ter duas notas finais diferentes na mesma turma. A relação que associa cada aluno de uma turma aos seus progenitores não é uma função: os alunos podem ter zero, ou dois, ou mais progenitores.

Podemos definir funções especificando de modo preciso qual é o valor da função para cada argumento possível. Formas de o fazer são dar uma fórmula, descrever um método para calcular o valor, ou listar os valores para cada argumento. Como quer que as funções sejam definidas, temos de garantir que, para cada argumento, especificamos um, e apenas um, valor.

Exemplo 3.5. Seja $f : \mathbb{N} \to \mathbb{N}$ definida por $f(x) = x + 1$. Esta é uma definição que especifica f como uma função que toma números naturais e devolve números naturais. Diz-nos que, dado um número natural x, f devolverá o seu sucessor $x + 1$. Neste caso, o contradomínio $\mathbb{N}$ não é a imagem de f, pois o número natural 0 não é sucessor de nenhum número natural. A imagem de f é o conjunto de todos os inteiros positivos, $\mathbb{Z}^+$.

Exemplo 3.6. Seja $g : \mathbb{N} \to \mathbb{N}$ definida por $g(x) = x + 2 - 1$. Isto nos diz que g é uma função que toma números naturais e devolve números naturais. Dado um número natural n, g devolverá o predecessor do sucessor do sucessor de n, isto é, $n + 1$.

Acabamos de considerar duas funções, f e g, com *definições* diferentes. Contudo, são a *mesma função.* Afinal, para qualquer número natural n, temos $f(n) = n + 1 = n + 2 - 1 = g(n)$. Em outras palavras: as nossas definições de f e g especificam o mesmo mapeamento por meio de equações diferentes. Implicitamente, pois, estamos nos apoiando num princípio de extensionalidade para funções,

$$\text{se } \forall x\, f(x) = g(x), \text{ então } f = g$$

desde que f e g compartilhem o mesmo domínio e contradomínio.

Exemplo 3.7. Também podemos definir funções por casos. Por exemplo, poderíamos definir $h : \mathbb{N} \to \mathbb{N}$ por

$$h(x) = \begin{cases} \frac{x}{2} & \text{se } x \text{ é par} \\ \frac{x+1}{2} & \text{se } x \text{ é ímpar.} \end{cases}$$

Como todo número natural é par ou ímpar, a saída desta função será sempre um número natural. Basta lembrar que, se se define uma função por casos, cada entrada possível tem de cair em exatamente um caso. Em alguns casos, isto exigirá uma prova de que os casos são exaustivos e exclusivos.

3.2 Tipos de Funções

Será útil introduzir uma espécie de taxonomia para alguns dos tipos de funções que encontramos com mais frequência.

Para começar, podemos querer considerar funções com a propriedade de que todo *membro* do contradomínio é um valor da função. Tais funções chamam-se sobrejetoras, e podem ser esquematizadas como na Figura 3.2.

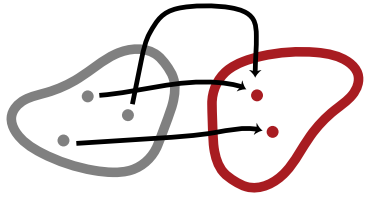


Figura 3.2: Uma função sobrejetora tem todo elemento do contradomínio como valor.

Definição 3.8 (Função Sobrejetora). Uma função $f : A \to B$ é *sobrejetora* se e só se B é também a imagem de f, isto é, para todo $y \in B$ existe pelo menos um $x \in A$ tal que $f(x) = y$, ou em símbolos:

$$(\forall y \in B)(\exists x \in A) f(x) = y.$$

Chamamos a tal função uma sobrejeção de A para B.

Se quiser mostrar que f é uma sobrejeção, tem de mostrar que todo objeto no contradomínio de f é o valor de $f(x)$ para alguma entrada x.

Note-se que qualquer função *induz* uma sobrejeção. Afinal, dada uma função $f : A \to B$, seja $f' : A \to \text{Im}(f)$ definida por $f'(x) = f(x)$. Como $\text{Im}(f)$ é *definida* como $\{f(x) \in B : x \in A\}$, esta função f' é garantidamente uma sobrejeção

Agora, qualquer função mapeia cada entrada possível para uma única saída. Mas há também funções que nunca mapeiam entradas diferentes para a mesma saída. Tais funções chamam-se injetoras, e podem ser esquematizadas como na Figura 3.3.

Figura 3.3: Uma função injetora nunca mapeia dois argumentos diferentes para o mesmo valor.

Definição 3.9 (Função Injetora). Uma função $f : A \to B$ é *injetora* se e só se, para cada $y \in B$, há no máximo um $x \in A$ tal que $f(x) = y$. Chamamos a tal função uma injeção de A para B.

Se quiser mostrar que f é uma injeção, tem de mostrar que, para quaisquer elementos x e y do domínio de f, se $f(x) = f(y)$, então $x = y$.

Exemplo 3.10. A função constante $f : \mathbb{N} \to \mathbb{N}$ dada por $f(x) = 1$ não é nem injetora, nem sobrejetora.

A função identidade $f : \mathbb{N} \to \mathbb{N}$ dada por $f(x) = x$ é tanto injetora quanto sobrejetora.

A função sucessora $f : \mathbb{N} \to \mathbb{N}$ dada por $f(x) = x + 1$ é injetora mas não sobrejetora.

A função $f : \mathbb{N} \to \mathbb{N}$ definida por:

$$f(x) = \begin{cases} \frac{x}{2} & \text{se } x \text{ é par} \\ \frac{x+1}{2} & \text{se } x \text{ é ímpar.} \end{cases}$$

é sobrejetora, mas não injetora.

Muitas vezes, queremos considerar funções que são tanto injetoras como sobrejetoras. Chamamos a tais funções bijetoras. Parecem-se com a função esquematizada na Figura 3.4. Bijeções são também por vezes chamadas *correspondências biunívocas*, pois associam de modo único elementos do contradomínio a elementos do domínio.

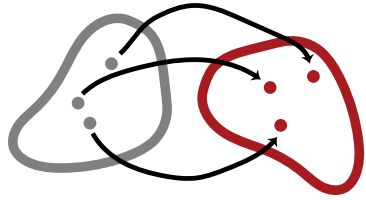


Figura 3.4: Uma função bijetora associa de modo único os elementos do contradomínio aos do domínio.

Definição 3.11 (Bijeção). Uma função $f : A \to B$ é *bijetora* se e só se é tanto sobrejetora como injetora. Chamamos a tal função uma bijeção de A para B (ou entre A e B).

3.3 Funções como Relações

Uma função que mapeia elementos de A para elementos de B define obviamente uma relação entre A e B, nomeadamente a relação que se verifica entre x e y se e só se $f(x) = y$. De fato, podemos até—se nos interessa reduzir os blocos de construção da matemática, por exemplo—*identificar* a função f com esta relação, isto é, com um conjunto de pares. Isto levanta então a questão: que relações definem funções deste modo?

Definição 3.12 (Gráfico de uma função). Seja $f : A \to B$ uma função. O *gráfico* de f é a relação $R_f \subseteq A \times B$ definida

por

$$R_f = \{\langle x, y\rangle : f(x) = y\}.$$

O gráfico de uma função fica unicamente determinado, pela extensionalidade. Além disso, a extensionalidade (sobre conjuntos) justifica de imediato o princípio implícito de extensionalidade para funções, segundo o qual, se f e g compartilham domínio e contradomínio, então são idênticas se coincidem em todos os valores.

Do mesmo modo, se uma relação é "funcional", então é o gráfico de uma função.

Proposição 3.13. *Seja $R \subseteq A \times B$ tal que:*

1. *Se Rxy e Rxz, então $y = z$; e*
2. *para todo $x \in A$ existe algum $y \in B$ tal que $\langle x, y\rangle \in R$.*

Então R é o gráfico da função $f : A \to B$ definida por $f(x) = y$ se e só se Rxy.

Prova. Suponhamos que existe y tal que Rxy. Se existisse outro $z \neq y$ tal que Rxz, a condição sobre R seria violada. Logo, se existe y tal que Rxy, esse y é único, e assim f está bem definida. Obviamente, $R_f = R$. □

Toda função $f : A \to B$ tem um gráfico, isto é, uma relação em $A \times B$ definida por $f(x) = y$. Por outro lado, toda relação $R \subseteq A \times B$ com as propriedades dadas na Proposição 3.13 é o gráfico de uma função $f : A \to B$. Por causa desta ligação estreita entre funções e os seus gráficos, podemos pensar numa função simplesmente como o seu gráfico. Em outras palavras, funções podem identificar-se com certas relações, isto é, com certos conjuntos de tuplos. Podemos agora considerar fazer operações semelhantes sobre funções às que fizemos sobre relações (ver seção 2.7). Em particular:

Definição 3.14. Seja $f : A \to B$ uma função com $C \subseteq A$.

A *restrição* de f a C é a função $f\restriction_C : C \to B$ definida por $(f\restriction_C)(x) = f(x)$ para todo $x \in C$. Em outras palavras, $f\restriction_C = \{\langle x,y\rangle \in R_f : x \in C\}$.

A *aplicação* de f a C é $f[C] = \{f(x) : x \in C\}$. Chamamos-lhe também a *imagem* de C por f.

Segue-se destas definições que $\mathrm{Im}(f) = f[\mathrm{dom}(f)]$, para qualquer função f. Estas noções são exatamente as que se esperaria, dadas as definições na seção 2.7 e a nossa identificação de funções com relações. Mas outras duas operações—inversas e produtos relativos—exigem um pouco mais de detalhe. Trataremos disso na seção 3.4 e na seção 3.5.

3.4 Inversas de Funções

Pensamos nas funções como mapas. Uma pergunta natural sobre funções é, pois, se o mapeamento pode ser "invertido". Por exemplo, a função sucessora $f(x) = x + 1$ pode ser invertida, no sentido em que a função $g(y) = y - 1$ "desfaz" o que f faz.

Mas é preciso ter cuidado. Embora a definição de g defina uma função $\mathbb{Z} \to \mathbb{Z}$, não define uma *função* $\mathbb{N} \to \mathbb{N}$, pois $g(0) \notin \mathbb{N}$. Assim, mesmo em casos simples, não é óbvio que uma função possa ser invertida; pode depender do domínio e do contradomínio.

Isto torna-se mais preciso com a noção de inversa de uma função.

Definição 3.15. Uma função $g : B \to A$ é uma *inversa* de uma função $f : A \to B$ se $f(g(y)) = y$ e $g(f(x)) = x$ para todo $x \in A$ e $y \in B$.

Se f tem uma inversa g, escrevemos muitas vezes f^{-1} em vez de g.

Agora determinaremos quando as funções têm inversas. Um bom candidato a inversa de $f\colon A \to B$ é $g\colon B \to A$ "definida por"

$$g(y) = \text{“o” } x \text{ tal que } f(x) = y.$$

Mas as aspas em "definida por" (e em "o") sugerem que isto não é uma definição. Pelo menos, nem sempre funcionará com total generalidade. Pois, para que esta definição especifique uma função, tem de haver um e apenas um x tal que $f(x) = y$—a saída de g tem de ficar unicamente determinada. Além disso, tem de estar especificada para todo $y \in B$. Se existirem x_1 e $x_2 \in A$ com $x_1 \neq x_2$ mas $f(x_1) = f(x_2)$, então $g(y)$ não ficaria unicamente determinada para $y = f(x_1) = f(x_2)$. E se não existir x algum tal que $f(x) = y$, então $g(y)$ nem sequer fica especificada. Em outras palavras, para g estar definida, f tem de ser tanto injetora quanto sobrejetora.

Vamos com calma. Dividiremos a questão em duas: dada uma função $f\colon A \to B$, quando existe uma função $g\colon B \to A$ tal que $g(f(x)) = x$? Tal g "desfaz" o que f faz, e chama-se *inversa à esquerda* de f. Em segundo lugar, quando existe uma função $h\colon B \to A$ tal que $f(h(y)) = y$? Tal h chama-se *inversa à direita* de f—f "desfaz" o que h faz.

Proposição 3.16. *Se $f\colon A \to B$ é injetora, então existe uma* inversa à esquerda *$g\colon B \to A$ de f tal que $g(f(x)) = x$ para todo $x \in A$.*

Prova. Suponhamos que $f\colon A \to B$ é injetora. Consideremos $y \in B$. Se $y \in \mathrm{Im}(f)$, existe $x \in A$ tal que $f(x) = y$. Como f é injetora, há apenas um tal $x \in A$. Então podemos definir: $g(y) = x$, isto é, $g(y)$ é "o" $x \in A$ tal que $f(x) = y$. Se $y \notin \mathrm{Im}(f)$, podemos mapeá-lo para qualquer $a \in A$. Assim, podemos escolher $a \in A$ e definir $g\colon B \to A$ por:

$$g(y) = \begin{cases} x & \text{se } f(x) = y \\ a & \text{se } y \notin \mathrm{Im}(f). \end{cases}$$

Está definida para todo $y \in B$, pois para cada tal $y \in \mathrm{Im}(f)$ há exatamente um $x \in A$ tal que $f(x) = y$. Por definição, se $y = f(x)$, então $g(y) = x$, isto é, $g(f(x)) = x$. □

Proposição 3.17. *Se $f\colon A \to B$ é sobrejetora, então existe uma* inversa à direita $h\colon B \to A$ *de f tal que $f(h(y)) = y$ para todo $y \in B$.*

Prova. Suponhamos que $f\colon A \to B$ é sobrejetora. Consideremos $y \in B$. Como f é sobrejetora, existe $x_y \in A$ com $f(x_y) = y$. Então podemos definir: $h(y) = x_y$, isto é, para cada $y \in B$ escolhemos algum $x \in A$ tal que $f(x) = y$; como f é sobrejetora há sempre pelo menos um de onde escolher.[1] Por definição, se $x = h(y)$, então $f(x) = y$, isto é, para todo $y \in B$, $f(h(y)) = y$. □

Combinando as ideias da prova anterior, obtemos agora que toda bijeção tem uma inversa, isto é, existe uma única função que é tanto inversa à esquerda quanto à direita de f.

Proposição 3.18. *Se $f\colon A \to B$ é bijetora, existe uma função $f^{-1}\colon B \to A$ tal que, para todo $x \in A$, $f^{-1}(f(x)) = x$ e, para todo $y \in B$, $f(f^{-1}(y)) = y$.*

Prova. Exercício. □

Há uma forma ligeiramente mais geral de extrair inversas. Vimos na seção 3.2 que toda função f induz uma sobrejeção $f'\colon A \to \mathrm{Im}(f)$ pondo $f'(x) = f(x)$ para todo $x \in A$. Claramente, se f é injetora, então f' é bijetora, logo tem uma inversa

[1] Como f é sobrejetora, para todo $y \in B$ o conjunto $\{x : f(x) = y\}$ é não vazio. A nossa definição de h exige que escolhamos um único x em cada um destes conjuntos. Que isto seja sempre possível não é de todo óbvio—a possibilidade de fazer estas escolhas é simplesmente assumida como axioma. Em outras palavras, esta proposição pressupõe o chamado Axioma da Escolha, questão a que aqui não damos destaque. Contudo, em muitos casos concretos, p.ex., quando $A = \mathbb{N}$ ou é finito, ou quando f é bijetora, o Axioma da Escolha não é necessário. (No caso em que f é bijetora, para cada $y \in B$ o conjunto $\{x : f(x) = y\}$ tem exatamente um elemento, logo não há escolha a fazer.)

única pela Proposição 3.18. Por um pequeno abuso de notação, chamamos por vezes à inversa de f' simplesmente "a inversa de f."

Proposição 3.19. *Mostre que, se $f : A \to B$ tem uma inversa à esquerda g e uma inversa à direita h, então $h = g$.*

Prova. Exercício. □

Proposição 3.20. *Toda função f tem no máximo uma inversa.*

Prova. Suponhamos que g e h são ambas inversas de f. Então, em particular, g é uma inversa à esquerda de f e h é uma inversa à direita. Pela Proposição 3.19, $g = h$. □

3.5 Composição de Funções

Vimos na seção 3.4 que a inversa f^{-1} de uma bijeção f é ela própria uma função. Outra operação sobre funções é a composição: podemos definir uma nova função compondo duas funções, f e g, isto é, aplicando primeiro f e depois g. Claro que isto só é possível se as imagens e os domínios forem compatíveis, isto é, a imagem de f tem de ser um subconjunto do domínio de g. Esta operação sobre funções é o análogo da operação de produto relativo sobre relações na seção 2.7.

Um diagrama pode ajudar a explicar a ideia de composição. Na Figura 3.5, representamos duas funções $f : A \to B$ e $g : B \to C$ e a sua composição $(g \circ f)$. A função $(g \circ f) : A \to C$ associa cada elemento de A a um elemento de C. Especificamos a que elemento de C fica associado um elemento de A do seguinte modo: dada uma entrada $x \in A$, aplicamos primeiro a função f a x, obtendo algum $f(x) = y \in B$, depois aplicamos a função g a y, obtendo algum $g(f(x)) = g(y) = z \in C$.

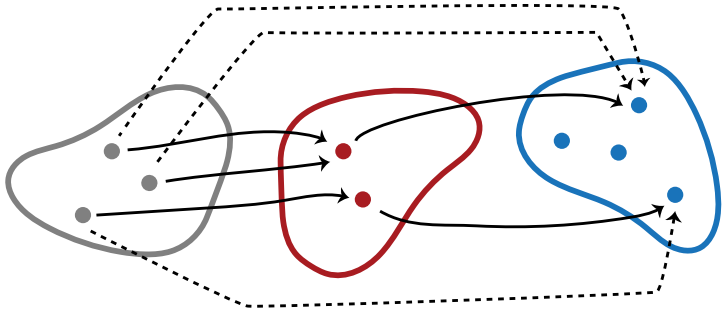


Figura 3.5: A composição $g \circ f$ de duas funções f e g.

Definição 3.21 (Composição). Sejam $f : A \to B$ e $g : B \to C$ funções. A *composição* de f com g é $g \circ f : A \to C$, onde $(g \circ f)(x) = g(f(x))$.

Exemplo 3.22. Consideremos as funções $f(x) = x + 1$, e $g(x) = 2x$. Como $(g \circ f)(x) = g(f(x))$, para cada entrada x deve primeiro tomar-se o sucessor, depois multiplicar o resultado por dois. Logo a composição é dada por $(g \circ f)(x) = 2(x + 1)$.

3.6 Funções Parciais

Às vezes é útil relaxar a definição de função de modo que não seja exigido que a saída da função esteja definida para todas as entradas possíveis. Tais mapeamentos são chamados de *funções parciais*.

Definição 3.23. Uma *função parcial* $f : A \nrightarrow B$ é um mapeamento que associa a cada elemento de A no máximo um elemento de B. Se f associa um elemento de B a $x \in A$, dizemos que $f(x)$ está *definido*, e caso contrário *indefinido*. Se $f(x)$ está definido, escrevemos $f(x) \downarrow$, caso contrário $f(x) \uparrow$. O *domínio* de uma

função parcial f é o subconjunto de A onde ela está definida, isto é, $\mathrm{dom}(f) = \{x \in A : f(x)\downarrow\}$.

Exemplo 3.24. Toda função $f : A \to B$ é também uma função parcial. Funções parciais que estão definidas em todo A—isto é, o que até agora chamávamos simplesmente de função—também são chamadas de funções *totais*.

Exemplo 3.25. A função parcial $f : \mathbb{R} \nrightarrow \mathbb{R}$ dada por $f(x) = 1/x$ é indefinida para $x = 0$ e definida em todo o restante.

Definição 3.26 (Gráfico de uma função parcial). Seja $f : A \nrightarrow B$ uma função parcial. O *gráfico* de f é a relação $R_f \subseteq A \times B$ definida por

$$R_f = \{\langle x, y\rangle : f(x) = y\}.$$

Proposição 3.27. *Suponha que $R \subseteq A \times B$ tem a propriedade de que sempre que Rxy e Rxy' então $y = y'$. Então R é o gráfico da função parcial $f : A \nrightarrow B$ definida por: se existe um y tal que Rxy, então $f(x) = y$, caso contrário $f(x)\uparrow$. Se R também é* serial, *isto é, para cada $x \in A$ existe um $y \in B$ tal que Rxy, então f é total.*

Prova. Suponha que existe um y tal que Rxy. Se existisse outro $y' \neq y$ tal que Rxy', a condição sobre R seria violada. Portanto, se existe um y tal que Rxy, esse y é único, e assim f é bem definida. Obviamente, $R_f = R$ e f é total se R é serial. □

Resumo

Uma **função** $f : A \to B$ mapeia todo elemento do **domínio** A em um único elemento do **contradomínio** B. Se $x \in A$, chamamos o y no qual f mapeia x de **valor** $f(x)$ de f para o **argumento** x. Se A é um conjunto de pares, podemos pensar na função f como

tomando dois argumentos. A **imagem** $\mathrm{Im}(f)$ de f é o subconjunto de B que consiste em todos os valores de f.

Se $\mathrm{Im}(f) = B$, então f é chamada de **sobrejetora**. O valor $f(x)$ é único no sentido de que f mapeia x em apenas um $f(x)$, nunca mais de um. Se $f(x)$ também é único no sentido de que dois argumentos diferentes não são mapeados no mesmo valor, f é chamada de **injetora**. As funções que são tanto injetoras quanto sobrejetoras são chamadas de **bijetoras**.

As funções bijetoras têm uma **função inversa** f^{-1} única. As funções também podem ser encadeadas: a função $(g \circ f)$ é a **composição** de f com g. Composições de funções injetoras são injetoras, e de funções sobrejetoras são sobrejetoras, e $(f^{-1} \circ f)$ é a função identidade.

Se relaxamos a exigência de que f deva ter um valor para todo $x \in A$, obtemos a noção de **função parcial**. Se $f : A \nrightarrow B$ é parcial, dizemos que $f(x)$ é **definida**, $f(x)\downarrow$, se f tem um valor para o argumento x, e, caso contrário, dizemos que $f(x)$ é **indefinida**, $f(x)\uparrow$. Toda função (parcial) f está associada ao **grafo** R_f de f, a relação que vale sse $f(x) = y$.

Problemas

Problema 3.1. Mostre que, se $f : A \to B$ tem uma inversa à esquerda g, então f é injetora.

Problema 3.2. Mostre que, se $f : A \to B$ tem uma inversa à direita h, então f é sobrejetora.

Problema 3.3. Prove Proposição 3.18. Tem de definir f^{-1}, mostrar que é uma função, e mostrar que é uma inversa de f, isto é, $f^{-1}(f(x)) = x$ e $f(f^{-1}(y)) = y$ para todo $x \in A$ e $y \in B$.

Problema 3.4. Prove Proposição 3.19.

Problema 3.5. Mostre que, se $f : A \to B$ e $g : B \to C$ são ambas injetora, então $g \circ f : A \to C$ é injetora.

Problema 3.6. Mostre que, se $f : A \to B$ e $g : B \to C$ são ambas sobrejetora, então $g \circ f : A \to C$ é sobrejetora.

Problema 3.7. Suponhamos $f : A \to B$ e $g : B \to C$. Mostre que o gráfico de $g \circ f$ é $R_f \mid R_g$.

Problema 3.8. Dada $f : A \nrightarrow B$, defina a função parcial $g : B \nrightarrow A$ por: para qualquer $y \in B$, se existe um único $x \in A$ tal que $f(x) = y$, então $g(y) = x$; caso contrário $g(y) \uparrow$. Mostre que se f é injetiva, então $g(f(x)) = x$ para todo $x \in \mathrm{dom}(f)$ e $f(g(y)) = y$ para todo $y \in \mathrm{Im}(f)$.

CAPÍTULO 4

O Tamanho dos Conjuntos

4.1 Introdução

Quando Georg Cantor desenvolveu a teoria dos conjuntos na década de 1870, um dos seus objetivos era tornar aceitável a ideia de uma coleção infinita—um infinito *atual*, como diriam os medievais. Parte central disto foi o seu tratamento do *tamanho* de conjuntos diferentes. Se a, b e c são todos distintos, então o conjunto $\{a, b, c\}$ é intuitivamente *maior* que $\{a, b\}$. Mas e os conjuntos infinitos? Serão todos do mesmo tamanho? Acontece que não.

A primeira ideia importante aqui é a de *enumeração*. Podemos listar qualquer conjunto finito enumerando todos os seus elementos. Para alguns conjuntos infinitos, também podemos listar todos os seus elementos se permitirmos que a própria lista seja infinita. Tais conjuntos chamam-se contáveis. O resultado surpreendente de Cantor, que compreenderemos completamente no fim deste capítulo, foi que alguns conjuntos infinitos *não* são contáveis.

4.2 Enumerações e Conjuntos Contáveis

Já demos exemplos de conjuntos listando seus elementos. Discutamos agora, de maneira mais geral, como e quando podemos listar os elementos de um conjunto, mesmo que esse conjunto seja infinito.

Definição 4.1 (Enumeração, informalmente). Informalmente, uma *enumeração* de um conjunto A é uma lista (possivelmente infinita) de elementos de A tal que todo elemento de A aparece na lista em alguma posição finita. Se A tem uma enumeração, então diz-se que A é *contável*.

Alguns pontos sobre enumerações:

1. Contamos como enumerações apenas listas que têm um começo e em que todo elemento além do primeiro tem um único elemento imediatamente anterior. Em outras palavras, há apenas um número finito de elementos entre o primeiro elemento da lista e qualquer outro elemento. Em particular, isso significa que todo elemento de uma enumeração tem uma posição finita: o primeiro elemento tem posição 1, o segundo posição 2, etc.

2. Podemos ter enumerações diferentes do mesmo conjunto A que diferem pela ordem em que os elementos aparecem: 4, 1, 25, 16, 9 enumera os (conjunto dos) cinco primeiros quadrados tão bem quanto 1, 4, 9, 16, 25.

3. Enumerações redundantes ainda são enumerações: 1, 1, 2, 2, 3, 3, ... enumera o mesmo conjunto que 1, 2, 3, ...

4. Ordem e redundância *importam* quando especificamos uma enumeração: podemos enumerar os inteiros positivos começando com 1, 2, 3, 1, ..., mas o padrão é mais fácil de ver quando enumerados da maneira padrão como 1, 2, 3, 4, ...

5. Enumerações devem ter um começo: $\ldots, 3, 2, 1$ não é uma enumeração dos inteiros positivos porque não tem primeiro elemento. Para ver como isso segue da definição informal, pergunte a si mesmo: "em que posição na lista o número 76 aparece?"

6. O seguinte não é uma enumeração dos inteiros positivos: $1, 3, 5, \ldots, 2, 4, 6, \ldots$ O problema é que os números pares ocorrem nas posições $\infty + 1$, $\infty + 2$, $\infty + 3$, em vez de em posições finitas.

7. O conjunto vazio é contável: é enumerado pela lista vazia!

Proposição 4.2. *Se A tem uma enumeração, tem uma enumeração sem repetições.*

Prova. Suponha que A tem uma enumeração $x_1, x_2, \ldots$ em que cada x_i é um elemento de A. Podemos remover repetições de uma enumeração removendo elementos repetidos. Por exemplo, podemos transformar a enumeração em uma nova na qual listamos x_i se ele é um elemento de A que não está entre $x_1, \ldots, x_{i-1}$ ou removemos x_i da lista se já aparece entre $x_1, \ldots, x_{i-1}$. □

O último argumento mostra que, para obter um bom controle sobre enumerações e conjuntos contáveis e provar coisas sobre eles, precisamos de uma definição mais precisa. O seguinte a fornece.

Definição 4.3 (Enumeração, formalmente). Uma *enumeração* de um conjunto $A \neq \emptyset$ é qualquer função sobrejetora $f : \mathbb{Z}^+ \to A$.

Convençamo-nos de que a definição formal e a definição informal usando uma lista possivelmente infinita são equivalentes. Primeiro, qualquer função sobrejetora de $\mathbb{Z}^+$ para um conjunto A enumera A. Tal função determina uma enumeração conforme definido informalmente acima: a lista $f(1), f(2), f(3), \ldots$. Como

f é sobrejetora, todo elemento de A é garantidamente o valor de $f(n)$ para algum $n \in \mathbb{Z}^+$. Logo, todo elemento de A aparece em alguma posição finita na lista. Como a função pode não ser injetora, a lista pode ser redundante, mas isso é aceitável (como observado acima).

Por outro lado, dada uma lista que enumera todos os elementos de A, podemos definir uma função sobrejetora $f: \mathbb{Z}^+ \to A$ fazendo $f(n)$ ser o n-ésimo elemento da lista, ou o último elemento da lista se não há n-ésimo elemento. O único caso em que isso não produz uma função sobrejetora é quando A é vazio, e portanto a lista é vazia. Assim, toda lista não vazia determina uma função sobrejetora $f: \mathbb{Z}^+ \to A$.

Definição 4.4. Um conjunto A é contável sse é vazio ou tem uma enumeração.

Exemplo 4.5. Uma função que enumera os inteiros positivos ($\mathbb{Z}^+$) é simplesmente a função identidade dada por $f(n) = n$. Uma função que enumera os números naturais $\mathbb{N}$ é a função $g(n) = n - 1$.

Exemplo 4.6. As funções $f: \mathbb{Z}^+ \to \mathbb{Z}^+$ e $g: \mathbb{Z}^+ \to \mathbb{Z}^+$ dadas por

$$f(n) = 2n \text{ e}$$
$$g(n) = 2n - 1$$

enumeram os inteiros positivos pares e os inteiros positivos ímpares, respectivamente. Contudo, nenhuma das funções é uma enumeração de $\mathbb{Z}^+$, já que nenhuma é sobrejetora.

Exemplo 4.7. A função $f(n) = (-1)^n \lceil \frac{(n-1)}{2} \rceil$ (onde $\lceil x \rceil$ denota a função *teto*, que arredonda x para cima até o inteiro mais próximo) enumera o conjunto dos inteiros $\mathbb{Z}$. Note como f gera os valores de $\mathbb{Z}$ "saltando" de um lado para outro entre inteiros

positivos e negativos:

$$\begin{array}{cccccccc} f(1) & f(2) & f(3) & f(4) & f(5) & f(6) & f(7) & \dots \\ -\lceil \frac{0}{2} \rceil & \lceil \frac{1}{2} \rceil & -\lceil \frac{2}{2} \rceil & \lceil \frac{3}{2} \rceil & -\lceil \frac{4}{2} \rceil & \lceil \frac{5}{2} \rceil & -\lceil \frac{6}{2} \rceil & \dots \\ 0 & 1 & -1 & 2 & -2 & 3 & \dots & \end{array}$$

Você também pode pensar em f como definida por casos da seguinte forma:

$$f(n) = \begin{cases} 0 & \text{se } n = 1 \\ n/2 & \text{se } n \text{ é par} \\ -(n-1)/2 & \text{se } n \text{ é ímpar e} > 1 \end{cases}$$

Embora seja talvez mais natural ao listar os elementos de um conjunto começar a contar a partir do 1º elemento, os matemáticos gostam de usar os números naturais $\mathbb{N}$ para contar coisas. Eles falam do 0º, 1º, 2º, e assim por diante, elemento de uma lista. Correspondentemente, podemos definir uma enumeração como uma função sobrejetora de $\mathbb{N}$ para A. É claro que as duas definições são equivalentes.

Proposição 4.8. *Há uma sobrejeção $f : \mathbb{Z}^+ \to A$ sse há uma sobrejeção $g : \mathbb{N} \to A$.*

Prova. Dada uma sobrejeção $f : \mathbb{Z}^+ \to A$, podemos definir $g(n) = f(n+1)$ para todo $n \in \mathbb{N}$. É fácil ver que $g : \mathbb{N} \to A$ é sobrejetora. Reciprocamente, dada uma sobrejeção $g : \mathbb{N} \to A$, defina $f(n) = g(n-1)$. □

Isso nos dá o seguinte resultado:

Corolário 4.9. *Um conjunto A é contável sse é vazio ou há uma função sobrejetora $f : \mathbb{N} \to A$.*

Discutimos acima que uma lista de elementos de um conjunto A pode ser transformada em uma lista sem repetições. Isso também vale para enumerações, mas é um pouco mais difícil de formular e provar rigorosamente. Qualquer função $f : \mathbb{Z}^+ \to A$ deve estar definida para todo $n \in \mathbb{Z}^+$. Se há apenas um número finito de elementos em A, então claramente não podemos ter uma função definida nos infinitos elementos de $\mathbb{Z}^+$ que toma como valores todos os elementos de A mas nunca toma o mesmo valor duas vezes. Nesse caso, isto é, no caso em que a lista sem repetições é finita, devemos escolher um domínio diferente para f, um com apenas um número finito de elementos. Não ter repetições significa que f deve ser injetora. Como também é sobrejetora, estamos procurando uma bijeção entre algum conjunto finito $\{1, \ldots, n\}$ ou $\mathbb{Z}^+$ e A.

Proposição 4.10. *Se $f : \mathbb{Z}^+ \to A$ é sobrejetora (isto é, uma enumeração de A), há uma bijeção $g : Z \to A$ onde Z é $\mathbb{Z}^+$ ou $\{1, \ldots, n\}$ para algum $n \in \mathbb{Z}^+$.*

Prova. Definimos a função g recursivamente: seja $g(1) = f(1)$. Se $g(i)$ já foi definido, seja $g(i+1)$ o primeiro valor de $f(1)$, $f(2)$, ... que ainda não está entre $g(1)$, ..., $g(i)$, se houver um. Se A tem apenas n elementos, então $g(1)$, ..., $g(n)$ estão todos definidos, e assim definimos uma função $g : \{1, \ldots, n\} \to A$. Se A tem infinitos elementos, então para qualquer i deve haver um elemento de A na enumeração $f(1)$, $f(2)$, ..., que ainda não está entre $g(1)$, ..., $g(i)$. Nesse caso definimos uma função $g : \mathbb{Z}^+ \to A$.

A função g é sobrejetora, já que qualquer elemento de A está entre $f(1)$, $f(2)$, ... (pois f é sobrejetora) e assim eventualmente será um valor de $g(i)$ para algum i. Também é injetora, já que se houvesse $j < i$ tal que $g(j) = g(i)$, então $g(i)$ já estaria entre $g(1)$, ..., $g(i-1)$, contrariamente à forma como definimos g. □

Corolário 4.11. *Um conjunto A é contável sse é vazio ou há uma bijeção $f : N \to A$ onde $N = \mathbb{N}$ ou $N = \{0, \ldots, n\}$ para algum $n \in \mathbb{N}$.*

Prova. A é contável sse A é vazio ou há a sobrejetora $f : \mathbb{Z}^+ \to A$. Pela Proposição 4.10, o último vale sse há uma função bijetora $f : Z \to A$ onde $Z = \mathbb{Z}^+$ ou $Z = \{1, \ldots, n\}$ para algum $n \in \mathbb{Z}^+$. Pelo mesmo argumento da prova da Proposição 4.8, isso por sua vez é o caso sse há uma bijeção $g : N \to A$ onde $N = \mathbb{N}$ ou $N = \{0, \ldots, n-1\}$. □

4.3 Método do Zigue-Zague de Cantor

Já consideramos algumas enumerações "fáceis". Vejamos algo um pouco mais difícil. Considere-se o conjunto dos pares de números naturais, que definimos na seção 1.5 assim:

$$\mathbb{N} \times \mathbb{N} = \{\langle n, m\rangle : n, m \in \mathbb{N}\}$$

Podemos organizar estes pares ordenados numa *matriz*, assim:

	0	**1**	**2**	**3**	...
0	$\langle 0,0\rangle$	$\langle 0,1\rangle$	$\langle 0,2\rangle$	$\langle 0,3\rangle$	...
1	$\langle 1,0\rangle$	$\langle 1,1\rangle$	$\langle 1,2\rangle$	$\langle 1,3\rangle$	...
2	$\langle 2,0\rangle$	$\langle 2,1\rangle$	$\langle 2,2\rangle$	$\langle 2,3\rangle$	...
3	$\langle 3,0\rangle$	$\langle 3,1\rangle$	$\langle 3,2\rangle$	$\langle 3,3\rangle$	...
⋮	⋮	⋮	⋮	⋮	⋱

Claramente, cada par ordenado em $\mathbb{N} \times \mathbb{N}$ aparece exatamente uma vez na matriz. Em particular, $\langle n, m\rangle$ aparece na n-ésima linha e na m-ésima coluna. Mas como organizar os elementos de tal matriz numa lista "unidimensional"? O padrão na matriz abaixo demonstra uma forma de o fazer (embora haja, claro,

muitas outras):

	0	**1**	**2**	**3**	**4**	...
0	0	1	3	6	10	...
1	2	4	7	11	...	...
2	5	8	12	...	...	...
3	9	13	...	...	...	...
4	14	...	...	...	...	...
⋮	⋮	⋮	⋮	⋮	...	⋱

Este padrão chama-se *método do zigue-zague de Cantor*. Enumera $\mathbb{N} \times \mathbb{N}$ do seguinte modo:

$$\langle 0,0\rangle, \langle 0,1\rangle, \langle 1,0\rangle, \langle 0,2\rangle, \langle 1,1\rangle, \langle 2,0\rangle, \langle 0,3\rangle, \langle 1,2\rangle, \langle 2,1\rangle, \langle 3,0\rangle, \ldots$$

Isto estabelece o seguinte:

Proposição 4.12. *$\mathbb{N} \times \mathbb{N}$ é contável.*

Prova. Seja $f : \mathbb{N} \to \mathbb{N} \times \mathbb{N}$ a função que a cada $k \in \mathbb{N}$ associa o tuplo $\langle n, m\rangle \in \mathbb{N} \times \mathbb{N}$ tal que k é o valor na n-ésima linha e m-ésima coluna da matriz do zigue-zague de Cantor. □

Esta técnica generaliza-se bem. Por exemplo, podemos usá-la para enumerar o conjunto dos triplos ordenados de números naturais, isto é:

$$\mathbb{N} \times \mathbb{N} \times \mathbb{N} = \{\langle n, m, k\rangle : n, m, k \in \mathbb{N}\}$$

Pensamos em $\mathbb{N}\times\mathbb{N}\times\mathbb{N}$ como o produto cartesiano de $\mathbb{N}\times\mathbb{N}$ com $\mathbb{N}$, isto é,

$$\mathbb{N}^3 = (\mathbb{N} \times \mathbb{N}) \times \mathbb{N} = \{\langle\langle n, m\rangle, k\rangle : n, m, k \in \mathbb{N}\}$$

e assim podemos enumerar $\mathbb{N}^3$ com uma matriz rotulando um eixo com a enumeração de $\mathbb{N}$ e o outro com a enumeração de $\mathbb{N}^2$:

	0	**1**	**2**	**3**	...
$\langle \mathbf{0},\mathbf{0}\rangle$	$\langle 0,0,0\rangle$	$\langle 0,0,1\rangle$	$\langle 0,0,2\rangle$	$\langle 0,0,3\rangle$	...
$\langle \mathbf{0},\mathbf{1}\rangle$	$\langle 0,1,0\rangle$	$\langle 0,1,1\rangle$	$\langle 0,1,2\rangle$	$\langle 0,1,3\rangle$	...
$\langle \mathbf{1},\mathbf{0}\rangle$	$\langle 1,0,0\rangle$	$\langle 1,0,1\rangle$	$\langle 1,0,2\rangle$	$\langle 1,0,3\rangle$	...
$\langle \mathbf{0},\mathbf{2}\rangle$	$\langle 0,2,0\rangle$	$\langle 0,2,1\rangle$	$\langle 0,2,2\rangle$	$\langle 0,2,3\rangle$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

Assim, com um método como o zigue-zague de Cantor, obtemos analogamente uma enumeração de $\mathbb{N}^3$. E podemos continuar, obtendo enumerações de $\mathbb{N}^n$ para qualquer número natural n. Logo:

Proposição 4.13. *$\mathbb{N}^n$ é contável, para todo $n \in \mathbb{N}$.*

4.4 Funções de Pareamento e Códigos

O método do zigue-zague de Cantor torna visualmente evidente a enumerabilidade de $\mathbb{N}^n$. Mas vamos concentrar-nos na nossa matriz que representa $\mathbb{N}^2$. Seguindo a linha em zigue-zague na matriz e contando as posições, podemos verificar que $\langle 1,2\rangle$ fica associado ao número 7. Seria útil, contudo, poder calcular isto de modo mais direto. Ou seja, seria útil dispor da *inversa* da enumeração em zigue-zague, $g\colon \mathbb{N}^2 \to \mathbb{N}$, tal que

$$g(\langle 0,0\rangle) = 0,\ g(\langle 0,1\rangle) = 1,\ g(\langle 1,0\rangle) = 2,\ \ldots, g(\langle 1,2\rangle) = 7,\ \ldots$$

Isto permitir-nos-ia calcular exatamente em que posição da enumeração cai $\langle n,m\rangle$.

De fato, podemos definir g diretamente com base em duas observações. Primeiro: se a n-ésima linha e a m-ésima coluna contêm o valor v, então a $(n+1)$-ésima linha e a $(m-1)$-ésima coluna contêm o valor $v+1$. Segundo: a primeira linha da nossa enumeração consiste nos números triangulares, começando por

0, 1, 3, 6, etc. O k-ésimo número triangular é a soma dos números naturais $< k$, que se pode calcular como $k(k+1)/2$. Juntando estas duas observações, consideremos a função:

$$g(n,m) = \frac{(n+m+1)(n+m)}{2} + n$$

Escreve-se frequentemente $g(n,m)$ em vez de $g(\langle n,m\rangle)$, já que é mais fácil para os olhos. Isto diz-nos que se determine primeiro o $(n+m)$-ésimo número triangular e depois que lhe somemos n. A função preenche a matriz exatamente da maneira que desejamos. Em particular, o par $\langle 1,2\rangle$ é enviado para $\frac{4\times 3}{2} + 1 = 7$.

Esta função g é a *inversa* de uma enumeração de um conjunto de pares. Tais funções chamam-se *funções de emparelhamento.*

Definição 4.14 (Função de emparelhamento). Uma função $f\colon A \times B \to \mathbb{N}$ é uma *função de emparelhamento* aritmética se f for injetiva. Dizemos também que f *codifica* $A \times B$, e que $f(x,y)$ é o *código* de $\langle x,y\rangle$.

Podemos usar funções de emparelhamento para codificar, por exemplo, pares de números naturais; ou, em outras palavras, podemos representar cada *par* de elementos por um *único* número. Usando a inversa da função de emparelhamento, podemos *descodificar* o número, isto é, determinar que par representa.

4.5 Uma Função de Emparelhamento Alternativa

Há outras enumerações de $\mathbb{N}^2$ que facilitam descobrir quais são suas inversas. Eis uma. Em vez de visualizar a enumeração em uma matriz, comece com a lista de inteiros positivos associados a espaços (inicialmente) vazios. Imagine preencher esses espaços sucessivamente com pares $\langle n,m\rangle$ da seguinte forma. Começando com os pares que têm 0 na primeira posição (isto é, pares $\langle 0,m\rangle$),

coloque o primeiro (isto é, $\langle 0,0\rangle$) no primeiro espaço vazio, depois pule um espaço vazio, coloque o segundo (isto é, $\langle 0,2\rangle$) no próximo espaço vazio, pule outro, e assim por diante. O começo (incompleto) de nossa enumeração agora se parece com isto

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	...
$\langle 0,0\rangle$		$\langle 0,1\rangle$		$\langle 0,2\rangle$		$\langle 0,3\rangle$		$\langle 0,4\rangle$		...

Repita isso com pares $\langle 1,m\rangle$ nos lugares que ainda permanecem vazios, novamente pulando cada outro espaço vazio:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	...
$\langle 0,0\rangle$	$\langle 1,0\rangle$	$\langle 0,1\rangle$		$\langle 0,2\rangle$	$\langle 1,1\rangle$	$\langle 0,3\rangle$		$\langle 0,4\rangle$	$\langle 1,2\rangle$	...

Insira pares $\langle 2,m\rangle$, $\langle 2,m\rangle$, etc., da mesma forma. Nossa enumeração completa assim começa assim:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	...
$\langle 0,0\rangle$	$\langle 1,0\rangle$	$\langle 0,1\rangle$	$\langle 2,0\rangle$	$\langle 0,2\rangle$	$\langle 1,1\rangle$	$\langle 0,3\rangle$	$\langle 3,0\rangle$	$\langle 0,4\rangle$	$\langle 1,2\rangle$	...

Se numerarmos as células na matriz acima de acordo com essa enumeração, não encontraremos uma linha zigue-zague arrumada, mas este arranjo:

	0	**1**	**2**	**3**	**4**	**5**	...
0	1	3	5	7	9	11	...
1	2	6	10	14	18	...	...
2	4	12	20	28	...	...	...
3	8	24	40	...	...	...	...
4	16	48	...	...	...	...	...
5	32	...	...	...	...	...	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱

Podemos ver que os pares na linha 0 estão nas posições ímpares de nossa enumeração, isto é, o par $\langle 0,m\rangle$ está na posição $2m+1$; os pares na segunda linha, $\langle 1,m\rangle$, estão em posições cujo número

é o dobro de um número ímpar, especificamente, $2 \cdot (2m+1)$; os pares na terceira linha, $\langle 2, m\rangle$, estão em posições cujo número é quatro vezes um número ímpar, $4 \cdot (2m+1)$; e assim por diante. Os fatores de $(2m+1)$ para cada linha, $1, 2, 4, 8, \ldots$, são exatamente as potências de 2: $1 = 2^0$, $2 = 2^1$, $4 = 2^2$, $8 = 2^3$, $\ldots$ De fato, o expoente relevante é sempre o primeiro membro do par em questão. Assim, para o par $\langle n, m\rangle$ o fator é 2^n. Isso nos dá a fórmula geral: $2^n \cdot (2m+1)$. Contudo, isso é um mapeamento de pares para inteiros *positivos*, isto é, $\langle 0,0\rangle$ tem posição 1. Se quisermos começar na posição 0 devemos subtrair 1 do resultado. Isso nos dá:

Exemplo 4.15. A função $h\colon \mathbb{N}^2 \to \mathbb{N}$ dada por

$$h(n,m) = 2^n(2m+1) - 1$$

é uma função de emparelhamento para o conjunto de pares de números naturais $\mathbb{N}^2$.

De acordo com nossa segunda enumeração de $\mathbb{N}^2$, o par $\langle 0,0\rangle$ tem código $h(0,0) = 2^0(2 \cdot 0 + 1) - 1 = 0$; $\langle 1,2\rangle$ tem código $2^1 \cdot (2 \cdot 2+1) - 1 = 2 \cdot 5 - 1 = 9$; $\langle 2,6\rangle$ tem código $2^2 \cdot (2 \cdot 6+1) - 1 = 51$.

Às vezes basta codificar pares de números naturais $\mathbb{N}^2$ sem exigir que a codificação seja sobrejetora. Tais codificações têm inversas que são apenas funções parciais.

Exemplo 4.16. A função $j\colon \mathbb{N}^2 \to \mathbb{N}^+$ dada por

$$j(n,m) = 2^n 3^m$$

é uma função injetora $\mathbb{N}^2 \to \mathbb{N}$.

4.6 Conjuntos Incontáveis

Alguns conjuntos, como o conjunto $\mathbb{Z}^+$ dos inteiros positivos, são infinitos. Até agora vimos exemplos de conjuntos infinitos que eram todos contável. Contudo, há também conjuntos infinitos

que não têm essa propriedade. Tais conjuntos são chamados de *incontável*.

Em primeiro lugar, talvez já seja surpreendente que existam conjuntos incontável. Para qualquer conjunto contável A há uma função sobrejetora $f : \mathbb{Z}^+ \to A$. Se um conjunto é incontável, não há tal função. Isto é, nenhuma função mapeando os infinitos elementos de $\mathbb{Z}^+$ para A pode esgotar todo A. Assim, há "mais" elementos em A do que os infinitos inteiros positivos.

Como se provaria que um conjunto é incontável? É preciso mostrar que nenhuma tal função sobrejetora pode existir. Equivalentemente, é preciso mostrar que os elementos de A não podem ser enumerados em uma lista infinita unidirecional. A melhor maneira de fazer isso é mostrar que toda lista de elementos de A deve deixar de fora pelo menos um elemento; ou que nenhuma função $f : \mathbb{Z}^+ \to A$ pode ser sobrejetora. Podemos fazer isso usando o *método diagonal* de Cantor. Dada uma lista de elementos de A, digamos, x_1, x_2, ..., construímos outro elemento de A que, por sua construção, não pode estar naquela lista.

Nosso primeiro exemplo é o conjunto $\mathbb{B}^\omega$ de todas as sequências infinitas, sem lacunas, de 0s e 1s.

Teorema 4.17. *$\mathbb{B}^\omega$ é incontável.*

Prova. Suponha, por absurdo, que $\mathbb{B}^\omega$ é contável, isto é, suponha que há uma lista s_1, s_2, s_3, s_4, ... de todos os elementos de $\mathbb{B}^\omega$. Cada um desses s_i é em si uma sequência infinita de 0s e 1s. Chamemos o j-ésimo elemento da i-ésima sequência nessa lista de $s_i(j)$. Então a i-ésima sequência s_i é

$$s_i(1), s_i(2), s_i(3), \ldots$$

Podemos organizar essa lista, e os elementos de cada sequência s_i nela, em uma matriz:

	1	2	3	4	...
1	$\mathbf{s_1(1)}$	$s_1(2)$	$s_1(3)$	$s_1(4)$	...
2	$s_2(1)$	$\mathbf{s_2(2)}$	$s_2(3)$	$s_2(4)$	...
3	$s_3(1)$	$s_3(2)$	$\mathbf{s_3(3)}$	$s_3(4)$	...
4	$s_4(1)$	$s_4(2)$	$s_4(3)$	$\mathbf{s_4(4)}$	...
⋮	⋮	⋮	⋮	⋮	⋱

Os rótulos ao lado indicam o número da sequência na lista s_1, s_2, ...; os números no topo rotulam os elementos das sequências individuais. Por exemplo, $s_1(1)$ é um nome para qualquer número, um 0 ou um 1, que é o primeiro elemento na sequência s_1, e assim por diante.

Agora construímos uma sequência infinita, $\overline{s}$, de 0s e 1s que não pode estar nessa lista. A definição de $\overline{s}$ dependerá da lista s_1, s_2, ... Qualquer lista infinita de sequências infinitas de 0s e 1s dá origem a uma sequência infinita $\overline{s}$ que está garantidamente fora da lista.

Para definir $\overline{s}$, especificamos todos os seus elementos, isto é, especificamos $\overline{s}(n)$ para todo $n \in \mathbb{Z}^+$. Fazemos isso lendo a diagonal da matriz acima (daí o nome "método diagonal") e depois trocando cada 1 por 0 e cada 0 por 1. De maneira mais abstrata, definimos $\overline{s}(n)$ como 0 ou 1 conforme o n-ésimo elemento da diagonal, $s_n(n)$, é 1 ou 0.

$$\overline{s}(n) = \begin{cases} 1 & \text{se } s_n(n) = 0 \\ 0 & \text{se } s_n(n) = 1. \end{cases}$$

Se você prefere fórmulas a definições por casos, também pode definir $\overline{s}(n) = 1 - s_n(n)$.

Claramente $\overline{s}$ é uma sequência infinita de 0s e 1s, já que é apenas a sequência espelhada da sequência de 0s e 1s que aparecem na diagonal de nossa matriz. Assim $\overline{s}$ é um elemento de $\mathbb{B}^\omega$. Mas não pode estar na lista s_1, s_2, ... Por quê?

Não pode ser a primeira sequência na lista, s_1, porque difere de s_1 no primeiro elemento. Seja qual for $s_1(1)$, definimos $\overline{s}(1)$ como o oposto. Não pode ser a segunda sequência na lista, porque $\overline{s}$ difere de s_2 no segundo elemento: se $s_2(2)$ é 0, $\overline{s}(2)$ é 1, e vice-versa. E assim por diante.

Mais precisamente: se $\overline{s}$ estivesse na lista, haveria algum k tal que $\overline{s} = s_k$. Duas sequências são idênticas sse concordam em toda posição, isto é, para qualquer n, $\overline{s}(n) = s_k(n)$. Em particular, tomando $n = k$ como caso especial, $\overline{s}(k) = s_k(k)$ teria de valer. $s_k(k)$ é 0 ou 1. Se é 0, então $\overline{s}(k)$ deve ser 1—é assim que definimos $\overline{s}$. Mas se $s_k(k) = 1$, de novo por causa da forma como definimos $\overline{s}$, $\overline{s}(k) = 0$. Em ambos os casos $\overline{s}(k) \neq s_k(k)$.

Começamos supondo que há uma lista de elementos de $\mathbb{B}^\omega$, s_1, s_2, ... A partir dessa lista construímos uma sequência $\overline{s}$ que provamos não poder estar na lista. Mas ela definitivamente é uma sequência de 0s e 1s se todos os s_i são sequências de 0s e 1s, isto é, $\overline{s} \in \mathbb{B}^\omega$. Isso mostra em particular que não pode haver lista de *todos* os elementos de $\mathbb{B}^\omega$, já que para qualquer tal lista também poderíamos construir uma sequência $\overline{s}$ garantidamente fora da lista, de modo que a suposição de que há uma lista de todas as sequências em $\mathbb{B}^\omega$ leva a uma contradição. □

Esse método de prova é chamado de "diagonalização" porque usa a diagonal da matriz para definir $\overline{s}$. A diagonalização não precisa envolver a presença de uma matriz: podemos mostrar que conjuntos não são contável usando uma ideia semelhante mesmo quando não há matriz nem diagonal propriamente dita.

Teorema 4.18. *$\wp(\mathbb{Z}^+)$ não é contável.*

Prova. Procedemos da mesma forma, mostrando que para toda lista de subconjuntos de $\mathbb{Z}^+$ há um subconjunto de $\mathbb{Z}^+$ que não pode estar na lista. Suponha que a seguinte é uma lista dada de subconjuntos de $\mathbb{Z}^+$:

$$Z_1, Z_2, Z_3, \ldots$$

Agora definimos um conjunto $\overline{Z}$ tal que para qualquer $n \in \mathbb{Z}^+$, $n \in \overline{Z}$ sse $n \notin Z_n$:

$$\overline{Z} = \{n \in \mathbb{Z}^+ : n \notin Z_n\} \qquad \square$$

$\overline{Z}$ é claramente um conjunto de inteiros positivos, já que por suposição cada Z_n o é, e assim $\overline{Z} \in \wp(\mathbb{Z}^+)$. Mas $\overline{Z}$ não pode estar na lista. Para mostrar isso, estabeleceremos que para cada $k \in \mathbb{Z}^+$, $\overline{Z} \neq Z_k$.

Seja $k \in \mathbb{Z}^+$ arbitrário. Definimos $\overline{Z}$ de modo que para qualquer $n \in \mathbb{Z}^+$, $n \in \overline{Z}$ sse $n \notin Z_n$. Em particular, tomando $n = k$, $k \in \overline{Z}$ sse $k \notin Z_k$. Mas isso mostra que $\overline{Z} \neq Z_k$, já que k é um elemento de um mas não do outro, e assim $\overline{Z}$ e Z_k têm elementos diferentes. Como k foi arbitrário, $\overline{Z}$ não está na lista $Z_1, Z_2, \ldots$

A prova precedente não mencionou uma diagonal, mas você pode pensar nela como envolvendo uma diagonal se a imaginar assim: imagine os conjuntos $Z_1, Z_2, \ldots$, escritos em uma matriz, onde cada elemento $j \in Z_i$ é listado na j-ésima coluna. Digamos que os primeiros quatro conjuntos nessa lista são $\{1,2,3,\ldots\}$, $\{2,4,6,\ldots\}$, $\{1,2,5\}$ e $\{3,4,5,\ldots\}$. Então a matriz começaria com

$$\begin{array}{lccccccc}
Z_1 = \{ & \mathbf{1}, & 2, & 3, & 4, & 5, & 6, & \ldots\} \\
Z_2 = \{ & & \mathbf{2}, & & 4, & & 6, & \ldots\} \\
Z_3 = \{ & 1, & 2, & & & 5 & & \} \\
Z_4 = \{ & & & 3, & \mathbf{4}, & 5, & 6, & \ldots\} \\
\vdots & & & & & \ddots & &
\end{array}$$

Então $\overline{Z}$ é o conjunto obtido percorrendo a diagonal, deixando de fora quaisquer números que aparecem ao longo da diagonal e incluindo aqueles j onde a matriz tem uma lacuna na j-ésima linha/coluna. No caso acima, deixaríamos de fora 1 e 2, incluiríamos 3, deixaríamos de fora 4, etc.

4.7 Redução

Mostramos $\wp(\mathbb{Z}^+)$ como incontável por um argumento de diagonalização. Já tínhamos uma prova de que $\mathbb{B}^\omega$, o conjunto de todas as sequências infinitas de 0s e 1s, é incontável. Eis outra maneira de provar que $\wp(\mathbb{Z}^+)$ é incontável: mostre que *se* $\wp(\mathbb{Z}^+)$ *é contável, então* $\mathbb{B}^\omega$ *também é contável.* Como sabemos que $\mathbb{B}^\omega$ não é contável, $\wp(\mathbb{Z}^+)$ também não pode ser. Isso é chamado de *reduzir* um problema a outro—neste caso, reduzimos o problema de enumerar $\mathbb{B}^\omega$ ao problema de enumerar $\wp(\mathbb{Z}^+)$. Uma solução para o último—uma enumeração de $\wp(\mathbb{Z}^+)$—daria uma solução para o primeiro—uma enumeração de $\mathbb{B}^\omega$.

Como reduzimos o problema de enumerar um conjunto B ao de enumerar um conjunto A? Fornecemos uma maneira de transformar uma enumeração de A em uma enumeração de B. A maneira mais fácil de fazer isso é definir uma função sobrejetora $f : A \to B$. Se $x_1, x_2, \ldots$ enumera A, então $f(x_1), f(x_2), \ldots$ enumeraria B. Em nosso caso, procuramos uma função sobrejetora $f : \wp(\mathbb{Z}^+) \to \mathbb{B}^\omega$.

Prova do Teorema 4.18 por redução. Suponha que $\wp(\mathbb{Z}^+)$ fosse contável, e portanto que há uma enumeração dele, $Z_1, Z_2, Z_3, \ldots$

Defina a função $f : \wp(\mathbb{Z}^+) \to \mathbb{B}^\omega$ fazendo $f(Z)$ ser a sequência s_k tal que $s_k(n) = 1$ sse $n \in Z$, e $s_k(n) = 0$ caso contrário. Isso define claramente uma função, já que sempre que $Z \subseteq \mathbb{Z}^+$, qualquer $n \in \mathbb{Z}^+$ ou é um elemento de Z ou não é. Por exemplo, o conjunto $2\mathbb{Z}^+ = \{2, 4, 6, \ldots\}$ dos pares positivos é mapeado para a sequência $010101\ldots$, o conjunto vazio é mapeado para $0000\ldots$ e o conjunto $\mathbb{Z}^+$ para $1111\ldots$.

Também é sobrejetora: toda sequência de 0s e 1s corresponde a algum conjunto de inteiros positivos, nomeadamente aquele que tem como membros os inteiros correspondentes às posições onde a sequência tem 1s. Mais precisamente, suponha $s \in \mathbb{B}^\omega$. Defina $Z \subseteq \mathbb{Z}^+$ por:

$$Z = \{n \in \mathbb{Z}^+ : s(n) = 1\}$$

Então $f(Z) = s$, como pode ser verificado consultando a definição de f.

Agora considere a lista

$$f(Z_1), f(Z_2), f(Z_3), \ldots$$

Como f é sobrejetora, todo membro de $\mathbb{B}^\omega$ deve aparecer como valor de f para algum argumento, e assim deve aparecer na lista. Essa lista portanto deve enumerar todo $\mathbb{B}^\omega$.

Assim, se $\wp(\mathbb{Z}^+)$ fosse contável, $\mathbb{B}^\omega$ seria contável. Mas $\mathbb{B}^\omega$ é incontável (Teorema 4.17). Logo $\wp(\mathbb{Z}^+)$ é incontável. □

É fácil confundir a direção em que a redução vai. Por exemplo, uma função sobrejetora $g\colon \mathbb{B}^\omega \to B$ *não* estabelece que B é incontável. (Considere $g\colon \mathbb{B}^\omega \to \mathbb{B}$ definida por $g(s) = s(1)$, a função que mapeia uma sequência de 0s e 1s para seu primeiro elemento. Ela é sobrejetora, porque algumas sequências começam com 0 e outras começam com 1. Mas $\mathbb{B}$ é finito.) Note também que a função f deve ser sobrejetora, caso contrário o argumento não se sustenta: $f(x_1)$, $f(x_2)$, ... não estariam garantidos de incluir todos os elementos de B. Por exemplo,

$$h(n) = \underbrace{000\ldots0}_{n\ 0\text{s}}$$

define uma função $h\colon \mathbb{Z}^+ \to \mathbb{B}^\omega$, mas $\mathbb{Z}^+$ é contável.

4.8 Equinumerosidade

Temos uma noção intuitiva do "tamanho" dos conjuntos, que funciona bem para conjuntos finitos. E quanto aos infinitos? Se quisermos uma forma formal de comparar os tamanhos de dois conjuntos de *qualquer* cardinalidade, é boa ideia começar por definir quando dois conjuntos têm o mesmo tamanho. Eis Frege:

> Se um garçom quer ter certeza de que pôs exatamente tantas facas quantos pratos sobre a mesa, não precisa

> contar nem uns nem outros, se simplesmente puser uma faca à direita de cada prato, de modo que cada faca na mesa fique à direita de algum prato. Os pratos e as facas ficam assim correlacionados de modo único, e de fato por essa mesma relação espacial. (Frege, 1884, § 70)

A intuição central desta passagem traduz-se numa definição formal:

Definição 4.19. A é *equinumeroso* a B, escrevendo-se $A \approx B$, sse existir uma bijeção $f : A \to B$.

Proposição 4.20. *A relação de equinumerosidade é uma relação de equivalência.*

Prova. É preciso mostrar que a equinumerosidade é reflexiva, simétrica e transitiva. Sejam A, B e C conjuntos.

Reflexividade. A aplicação identidade $\mathrm{Id}_A : A \to A$, com $\mathrm{Id}_A(x) = x$ para todo $x \in A$, é uma bijeção. Logo $A \approx A$.

Simetria. Suponhamos $A \approx B$, isto é, que existe uma bijeção $f : A \to B$. Como f é bijetora, a inversa f^{-1} existe e também é bijetora. Logo $f^{-1} : B \to A$ é uma bijeção, pelo que $B \approx A$.

Transitividade. Suponhamos $A \approx B$ e $B \approx C$, isto é, que existem bijeções $f : A \to B$ e $g : B \to C$. Então a composição $g \circ f : A \to C$ é bijetora, logo $A \approx C$. □

Proposição 4.21. *Se $A \approx B$, então A é contável se e só se B o é.*

Prova. Suponhamos $A \approx B$, logo existe alguma bijeção $f : A \to B$, e suponhamos que A é contável. Então ou $A = \emptyset$ ou existe uma função sobrejetora $g : \mathbb{Z}^+ \to A$. Se $A = \emptyset$, então $B = \emptyset$ também (caso contrário existiria um elemento $y \in B$ sem $x \in A$ com $g(x) = y$). Se, por outro lado, $g : \mathbb{Z}^+ \to A$ for sobrejetora, então $f \circ g : \mathbb{Z}^+ \to B$ é sobrejetora. De fato, seja $y \in B$. Como f é

sobrejetora, existe $x \in A$ tal que $f(x) = y$. Como g é sobrejetora, existe $n \in \mathbb{Z}^+$ tal que $g(n) = x$. Logo,

$$(f \circ g)(n) = f(g(n)) = f(x) = y$$

e portanto $f \circ g$ é sobrejetora. Temos que $f \circ g$ é uma enumeração de B, logo B é contável.

Se B é contável, obtemos que A é contável repetindo o argumento com a bijeção $f^{-1}: B \to A$ em lugar de f. □

4.9 Conjuntos de Tamanhos Diferentes e o Teorema de Cantor

Apresentamos uma formulação precisa da ideia de que dois conjuntos têm o mesmo tamanho. Também podemos precisar a ideia de que um conjunto é menor que outro. A nossa definição de "é menor do que (ou equinumeroso)" exigirá, em vez de uma bijeção entre os conjuntos, uma injeção do primeiro para o segundo. Se tal função existir, o tamanho do primeiro conjunto é menor ou igual ao do segundo. Intuitivamente, uma injeção de um conjunto em outro garante que a imagem da função tem pelo menos tantos elementos quanto o domínio, pois não há dois elementos do domínio que vão parar ao mesmo elemento da imagem.

Definição 4.22. *A não é maior do que B*, escrevendo-se $A \preceq B$, sse existir uma injeção $f: A \to B$.

É claro que isto é uma relação reflexiva e transitiva, mas não simétrica (fica como exercício). Podemos ainda introduzir uma noção que diz que um conjunto é (estritamente) menor que outro.

Definição 4.23. *A é menor do que B*, escrevendo-se $A \prec B$, sse existir uma injeção $f: A \to B$ mas não existir bijeção $g: A \to B$, isto é, $A \preceq B$ e $A \not\approx B$.

É claro que esta relação é irreflexiva e transitiva. (Também fica como exercício.) Com esta notação, um conjunto A é contável sse $A \preceq \mathbb{N}$, e A é incontável sse $\mathbb{N} \prec A$. Isto permite reformular o Teorema 4.18 como a observação de que $\mathbb{Z}^+ \prec \wp(\mathbb{Z}^+)$. De fato, Cantor (1892) provou que este último ponto é *completamente geral*:

Teorema 4.24 (Cantor). *$A \prec \wp(A)$, para qualquer conjunto A.*

Prova. A aplicação $f(x) = \{x\}$ é uma injeção $f : A \to \wp(A)$, pois se $x \neq y$, então também $\{x\} \neq \{y\}$ por extensionalidade, logo $f(x) \neq f(y)$. Temos pois $A \preceq \wp(A)$.

Mostraremos agora que não pode existir função sobrejetora $g : A \to \wp(A)$, quanto mais bijetora, e portanto que $A \not\approx \wp(A)$. Suponhamos que $g : A \to \wp(A)$. Como g é total, todo $x \in A$ é mapeado para um subconjunto $g(x) \subseteq A$. Podemos mostrar que g não pode ser sobrejetora. Para isso, definimos um subconjunto $\overline{A} \subseteq A$ que, por definição, não pode estar no contradomínio de g. Seja

$$\overline{A} = \{x \in A : x \notin g(x)\}.$$

Como $g(x)$ está definido para todo $x \in A$, $\overline{A}$ é claramente um subconjunto bem definido de A. Mas não pode estar no contradomínio de g. Seja $x \in A$ arbitrário; mostramos que $\overline{A} \neq g(x)$. Se $x \in g(x)$, então não satisfaz $x \notin g(x)$, logo, pela definição de $\overline{A}$, temos $x \notin \overline{A}$. Se $x \in \overline{A}$, tem de satisfazer a propriedade definidora de $\overline{A}$, isto é, $x \in A$ e $x \notin g(x)$. Como x era arbitrário, isto mostra que, para cada $x \in A$, $x \in g(x)$ sse $x \notin \overline{A}$, logo $g(x) \neq \overline{A}$. Em outras palavras, $\overline{A}$ não pode estar no contradomínio de g, o que contradiz a suposição de que g é sobrejetora. □

É instrutivo comparar a prova do Teorema 4.24 com a do Teorema 4.18. Ali mostramos que, para qualquer lista Z_1, Z_2, ..., de subconjuntos de $\mathbb{Z}^+$, se pode construir um conjunto $\overline{Z}$ de números garantidamente fora da lista. Estava garantido que não estava na lista porque, para todo $n \in \mathbb{Z}^+$, $n \in Z_n$ sse $n \notin \overline{Z}$. Assim, há sempre algum número que é um elemento de um entre Z_n ou

$\overline{Z}$ mas não do outro. Seguimos a mesma ideia aqui, exceto que os índices n são agora elementos de A em vez de $\mathbb{Z}^+$. O conjunto $\overline{A}$ está definido de modo a diferir de $g(x)$ para cada $x \in A$, porque $x \in g(x)$ sse $x \notin \overline{A}$. De novo, há sempre um elemento de A que é um elemento de um entre $g(x)$ e $\overline{A}$ mas não do outro. E tal como $\overline{Z}$ portanto não pode estar na lista $Z_1, Z_2, \ldots$, $\overline{A}$ não pode estar no contradomínio de g.

A prova vale também a pena comparar com a do Paradoxo de Russell, Teorema 1.29. De fato, o Teorema de Cantor foi a inspiração para o paradoxo de Russell.

4.10 A Noção de Tamanho e Schröder-Bernstein

Eis um pensamento intuitivo: se A não é maior do que B e B não é maior do que A, então A e B são equinumerosos. Sejamos honestos: se isto estivesse *errado*, dificilmente poderíamos justificar que a noção definida de equinumerosidade tem a ver com comparações de "tamanhos" entre conjuntos. Felizmente, o pensamento intuitivo está certo. Isto justifica-se pelo Teorema de Schröder-Bernstein.

Teorema 4.25 (Schröder-Bernstein). *Se $A \preceq B$ e $B \preceq A$, então $A \approx B$.*

Em outras palavras, se existir uma injeção de A para B e uma injeção de B para A, então existe uma bijeção de A para B.

Este resultado é, contudo, realmente bastante *difícil* de provar. De fato, embora Cantor o tivesse enunciado, outros provaram-no.[1] Por agora, pode (e deve) aceitá-lo como verdadeiro.

Felizmente, Schröder-Bernstein é *correto* e legitima pensarmos nas relações que definimos, isto é, $A \approx B$ e $A \preceq B$, como tendo a ver com "tamanho". Além disso, Schröder-Bernstein é muito *útil*.

[1] Para mais sobre a história, ver por exemplo Potter (2004, pp. 165–6).

Pode ser difícil imaginar uma bijeção entre dois conjuntos equinumerosos. O Teorema de Schröder-Bernstein permite decompor a comparação em casos em que só precisamos de uma injeção do primeiro para o segundo e vice-versa.

Resumo

O tamanho de um conjunto A pode ser medido por um número natural se o conjunto é finito, e tamanhos podem ser comparados comparando esses números. Se os conjuntos são infinitos, as coisas são mais complicadas. O primeiro nível de infinitude é o dos conjuntos **infinitos contáveis**. Um conjunto A é contável se seus elementos podem ser dispostos em uma **enumeração**, uma lista infinita unidirecional, isto é, quando há uma função sobrejetora $f : \mathbb{Z}^+ \to A$. Ele é infinito contável se é contável mas não finito. O **método zigue-zague** de Cantor mostra que o conjunto de pares de elementos de conjuntos infinitos contáveis é também contável; e isso pode ser usado para mostrar que mesmo o conjunto dos números racionais $\mathbb{Q}$ é contável.

Há, no entanto, conjuntos infinitos que não são contáveis: esses conjuntos são chamados de **incontáveis**. Há duas maneiras de mostrar que um conjunto é incontável: diretamente, usando um **argumento diagonal**, ou por **redução**. Para dar um argumento diagonal, supomos que o conjunto A em questão é contável, e usamos uma enumeração hipotética para definir um elemento de A que, pelo próprio modo como o definimos, está garantidamente diferente de todo elemento na enumeração. Assim, a enumeração não pode, afinal, ser uma enumeração de todo o A, e mostramos que não pode existir enumeração alguma de A. Uma redução mostra que A é incontável associando todo elemento de A a um elemento de algum conjunto incontável conhecido B de modo sobrejetor. Se isso é possível, então uma enumeração hipotética de A produziria uma enumeração de B. Como B é incontável, não pode existir enumeração alguma de A.

Em geral, conjuntos infinitos podem ser comparados quanto ao tamanho: A e B têm o mesmo tamanho, ou são **equinumerosos**, se há uma bijeção entre eles. Também podemos definir que A não é maior que B ($A \preceq B$) se há uma função injetora de A para B. Pelo Teorema de Schröder-Bernstein, isso de fato fornece uma ordem de tamanho dos conjuntos infinitos. Por fim, o **teorema de Cantor** diz que, para qualquer A, $A \prec \wp(A)$. Isso é uma generalização do nosso resultado de que $\wp(\mathbb{Z}^+)$ é incontável, e mostra que há não apenas dois, mas infinitos níveis de infinitude.

Problemas

Problema 4.1. Defina uma enumeração dos quadrados positivos 1, 4, 9, 16, ...

Problema 4.2. Mostre que se A e B são contáveis, então $A \cup B$ também é. Para isso, suponha que há funções sobrejetoras $f\colon \mathbb{Z}^+ \to A$ e $g\colon \mathbb{Z}^+ \to B$, e defina uma função sobrejetora $h\colon \mathbb{Z}^+ \to A \cup B$ e mostre que ela é sobrejetora. Considere também os casos em que A ou $B = \emptyset$.

Problema 4.3. Mostre que se $B \subseteq A$ e A é contável, então B também é. Para isso, suponha que há uma função sobrejetora $f\colon \mathbb{Z}^+ \to A$. Defina uma função sobrejetora $g\colon \mathbb{Z}^+ \to B$ e mostre que ela é sobrejetora. O que acontece se $B = \emptyset$?

Problema 4.4. Mostre por indução em n que se A_1, A_2, ..., A_n são todos contáveis, então $A_1 \cup \cdots \cup A_n$ também é. Você pode assumir o fato de que se dois conjuntos A e B são contáveis, então $A \cup B$ também é.

Problema 4.5. De acordo com a Definição 4.4, um conjunto A é contável sse $A = \emptyset$ ou há a sobrejetora $f\colon \mathbb{Z}^+ \to A$. Também é possível definir "conjunto contável" precisamente por: um conjunto é contável sse há uma função injetora $g\colon A \to \mathbb{Z}^+$. Mostre

que as definições são equivalentes, isto é, mostre que há uma função injetora $g : A \to \mathbb{Z}^+$ sse $A = \emptyset$ ou há a sobrejetora $f : \mathbb{Z}^+ \to A$.

Problema 4.6. Mostre que $(\mathbb{Z}^+)^n$ é contável, para todo $n \in \mathbb{N}$.

Problema 4.7. Mostre que $(\mathbb{Z}^+)^*$ é contável. Pode assumir PROBLEMA 4.6.

Problema 4.8. Dê uma enumeração do conjunto de todos os números racionais não negativos.

Problema 4.9. Mostre que $\mathbb{Q}$ é contável. Recordemos que qualquer número racional se pode escrever como uma fração z/m com $z \in \mathbb{Z}$, $m \in \mathbb{N}^+$.

Problema 4.10. Defina uma enumeração de $\mathbb{B}^*$.

Problema 4.11. Recordando o curso introdutório de lógica, cada tabela de verdade possível exprime uma função de verdade. Em outras palavras, as funções de verdade são todas as funções de $\mathbb{B}^k \to \mathbb{B}$ para algum k. Prove que o conjunto de todas as funções de verdade é contável.

Problema 4.12. Mostre que o conjunto de todos os subconjuntos finitos de um conjunto infinito contável arbitrário é contável.

Problema 4.13. Um subconjunto de $\mathbb{N}$ diz-se *confinito* sse for o complemento de um conjunto finito; isto é, $A \subseteq \mathbb{N}$ é confinito sse $\mathbb{N} \setminus A$ for finito. Seja I o conjunto cujos elementos são exatamente os subconjuntos finitos e confinitos de $\mathbb{N}$. Mostre que I é contável.

Problema 4.14. Mostre que a união contável de conjuntos contáveis é contável. Isto é, sempre que $A_1, A_2, \ldots$ são conjuntos e cada A_i é contável, então a união $\bigcup_{i=1}^{\infty} A_i$ de todos eles também é contável. [Nota: isto é difícil!]

Problema 4.15. Seja $f : A \times B \to \mathbb{N}$ uma função de emparelhamento arbitrária. Mostre que a inversa de f é uma enumeração de $A \times B$.

Problema 4.16. Especifique uma função que codifique $\mathbb{N}^3$.

Problema 4.17. Mostre que $\wp(\mathbb{N})$ é incontável por um argumento diagonal.

Problema 4.18. Mostre que o conjunto de funções $f : \mathbb{Z}^+ \to \mathbb{Z}^+$ é incontável por um argumento diagonal explícito. Isto é, mostre que se f_1, f_2, ..., é uma lista de funções e cada $f_i : \mathbb{Z}^+ \to \mathbb{Z}^+$, então há alguma $\overline{f} : \mathbb{Z}^+ \to \mathbb{Z}^+$ fora dessa lista.

Problema 4.19. Mostre que se há uma função injetora $g : B \to A$, e B é incontável, então A também é. Faça isso mostrando como você pode usar g para transformar uma enumeração de A em uma de B.

Problema 4.20. Mostre que o conjunto de todos os *conjuntos de* pares de inteiros positivos é incontável por um argumento de redução.

Problema 4.21. Mostre que o conjunto X de todas as funções $f : \mathbb{N} \to \mathbb{N}$ é incontável por um argumento de redução (Dica: dê uma função sobrejetora de X para $\mathbb{B}^\omega$.)

Problema 4.22. Mostre que $\mathbb{N}^\omega$, o conjunto de sequências infinitas de números naturais, é incontável por um argumento de redução.

Problema 4.23. Seja P o conjunto de funções do conjunto dos inteiros positivos para o conjunto $\{0\}$, e seja Q o conjunto de funções *parciais* do conjunto dos inteiros positivos para o conjunto $\{0\}$. Mostre que P é contável e Q não é. (Dica: reduza o problema de enumerar $\mathbb{B}^\omega$ a enumerar Q).

Problema 4.24. Seja S o conjunto de todas as funções sobrejetoras do conjunto dos inteiros positivos para o conjunto $\{0,1\}$, isto é, S consiste de todas as sobrejetoras $f : \mathbb{Z}^+ \to \mathbb{B}$. Mostre que S é incontável.

Problema 4.25. Mostre que o conjunto $\mathbb{R}$ de todos os números reais é incontável.

Problema 4.26. Mostre que, se $A \approx C$ e $B \approx D$, e $A \cap B = C \cap D = \emptyset$, então $A \cup B \approx C \cup D$.

Problema 4.27. Mostre que, se A é infinito e contável, então $A \approx \mathbb{N}$.

Problema 4.28. Mostre que não pode existir uma injeção $g : \wp(A) \to A$, para qualquer conjunto A. Sugestão: suponha $g : \wp(A) \to A$ injetora. Considere $D = \{g(B) : B \subseteq A \text{ e } g(B) \notin B\}$. Seja $x = g(D)$. Use o fato de g ser injetora para obter uma contradição.

Kurt Gödel
1906 - 1978

PARTE II

Lógica de Primeira Ordem

CAPÍTULO 5

Introdução à Lógica de Primeira Ordem

5.1 Lógica de Primeira Ordem

Você provavelmente já conhece a lógica de primeira ordem da sua primeira introdução à lógica formal.[1] Pode conhecê-la como "lógica quantificacional" ou "lógica de predicados". A lógica de primeira ordem, antes de tudo, é uma linguagem formal. Isso significa que ela tem um certo vocabulário, e suas expressões são cadeias desse vocabulário. Mas nem toda cadeia é permitida. Há diferentes tipos de expressões permitidas: termos, fórmulas e sentenças. Estamos principalmente interessados em sentenças da lógica de primeira ordem: elas nos fornecem um análogo formal

[1] De fato, mais ou menos assumimos que sim! Se não conhece, pode revisar um livro-texto mais elementar, como *forall x* (Magnus et al., 2021).

de sentenças do inglês, e sobre elas podemos fazer as perguntas que um lógico tipicamente se interessa. Por exemplo:

- B segue logicamente de A?
- A é logicamente verdadeira, logicamente falsa ou contingente?
- A e B são equivalentes?

Essas perguntas são primariamente perguntas sobre o "significado" das sentenças da lógica de primeira ordem. Por exemplo, um filósofo analisaria a pergunta de se B segue logicamente de A como perguntando: há um caso em que A é verdadeira mas B é falsa (B não segue de A), ou todo caso que torna A verdadeira também torna B verdadeira (B segue de A)? Mas ainda não nos foi dito o que é um "caso"—esse é o trabalho da *semântica*. A semântica da lógica de primeira ordem fornece um modelo matematicamente preciso da ideia intuitiva do filósofo de "caso", e também—e isso é importante—do que significa uma sentença A ser *verdadeira em* um caso. Chamamos o modelo matematicamente preciso que desenvolveremos de uma estrutura. A relação que torna "verdadeira em" precisa é chamada de relação de *satisfação*. Assim, o que definiremos é "A é satisfeita em $\mathfrak{M}$" (em símbolos: $\mathfrak{M} \vDash A$) para sentenças A e estruturas $\mathfrak{M}$. Uma vez feito isso, também podemos dar definições precisas dos outros termos semânticos como "segue de" ou "é logicamente verdadeira". Essas definições tornarão possível decidir, novamente com precisão matemática, se, por exemplo, $\forall x\,(A(x) \to B(x)), \exists x\,A(x) \vDash \exists x\,B(x)$. A resposta será, é claro, "sim". Se você já foi treinado para simbolizar sentenças do inglês na lógica de primeira ordem, reconhecerá isto como, por exemplo, a simbolização de "Todas as formigas são insetos, há formigas, portanto há insetos." Isso é obviamente um argumento válido, e assim nosso modelo matemático de "segue de" para nossa linguagem formal deve dar a mesma resposta.

Outro tópico que você provavelmente lembra da sua primeira introdução à lógica formal é que há *derivações*. Se você fez um primeiro curso de lógica formal, seu professor o fez praticar a encontrar tais derivações, talvez até uma derivação que mostra que a consequência acima vale. Há muitas maneiras diferentes de dar derivações: você pode ter feito algo chamado "dedução natural" ou "árvores de verdade", mas há muitas outras. O propósito dos sistemas de derivação é fornecer ferramentas com as quais as perguntas dos lógicos acima podem ser respondidas: por exemplo, uma derivação de dedução natural em que $\forall x\,(A(x) \to B(x))$ e $\exists x\,A(x)$ são premissas e $\exists x\,B(x)$ é a conclusão (última linha) *verifica* que $\exists x\,B(x)$ segue logicamente de $\forall x\,(A(x) \to B(x))$ e $\exists x\,A(x)$.

Mas por quê? À primeira vista, os sistemas de derivação não têm nada a ver com semântica: dar uma derivação formal envolve apenas dispor símbolos de certas maneiras regidas por regras; eles não mencionam "casos" ou "verdadeira em" de forma alguma. A conexão entre sistemas de derivação e semântica tem de ser estabelecida por uma investigação metalógica. O que é necessário é uma prova matemática, por exemplo, de que uma derivação formal de $\exists x\,B(x)$ a partir das premissas $\forall x\,(A(x) \to B(x))$ e $\exists x\,A(x)$ é possível se, e somente se, $\exists x\,B(x)$ é consequência de $\forall x\,(A(x) \to B(x))$ e $\exists x\,A(x)$ em conjunto. Antes que isso possa ser feito, entretanto, muito trabalho minucioso tem de ser realizado para acertar as definições de sintaxe e semântica.

5.2 Sintaxe

Primeiro devemos tornar preciso quais cadeias de símbolos contam como sentenças da lógica de primeira ordem. Faremos isso mais adiante; por enquanto, procederemos por exemplos. Os blocos de construção básicos—o vocabulário—da lógica de primeira ordem dividem-se em duas partes. A primeira parte são os símbolos que usamos para dizer coisas específicas ou para destacar coisas específicas. Destacamos coisas usando símbolos

de constante, e dizemos coisas sobre o que destacamos usando predicados. Por exemplo, poderíamos usar a como um símbolo de constante para destacar uma única coisa, e então dizer algo sobre ela usando a sentença $P(a)$. Se você tem significados para "a" e "P" em mente, pode ler $P(a)$ como uma sentença do inglês (e provavelmente já fez isso quando aprendeu lógica formal pela primeira vez). Uma vez que se tem tais sentenças simples da lógica de primeira ordem, pode-se construir outras mais complexas usando a segunda parte do vocabulário: os símbolos lógicos (conectivos e quantificadores). Assim, por exemplo, podemos formar expressões como $(P(a) \wedge Q(b))$ ou $\exists x\, P(x)$.

Para fornecer as definições precisas de semântica e as regras dos nossos sistemas de derivação necessárias para o estudo metalógico rigoroso, temos primeiro que dar uma definição precisa do que conta como uma sentença da lógica de primeira ordem. A ideia básica é fácil o suficiente de entender: há algumas sentenças simples que podemos formar apenas a partir de predicados e símbolos de constante, como $P(a)$. E então a partir destas formamos outras mais complexas usando os conectivos e quantificadores. Mas quais são exatamente as regras pelas quais nos é permitido formar sentenças mais complexas? Elas devem ser especificadas; caso contrário, não definimos "sentença da lógica de primeira ordem" com precisão suficiente. Há algumas questões. A primeira é obter as cadeias certas para contarem como sentenças. A segunda é fazê-lo de tal forma que possamos dar provas matemáticas sobre *todas* as sentenças. Finalmente, teremos também que dar definições precisas de algumas operações rudimentares com sentenças, como "substituir todo x em A por b." O problema é que os quantificadores e variáveis que temos na lógica de primeira ordem tornam não inteiramente óbvio como isso deve ser feito. Por exemplo, $\exists x\, P(a)$ deve contar como uma sentença? E quanto a $\exists x\, \exists x\, P(x)$? Qual deveria ser o resultado de "substituir x por b em $(P(x) \wedge \exists x\, P(x))$"?

5.3 Fórmulas

Eis a abordagem que usaremos para especificar rigorosamente as sentenças da lógica de primeira ordem e para lidar com as questões decorrentes do uso de variáveis. Primeiro definimos um conjunto *diferente* de expressões: fórmulas. Uma vez feito isso, podemos considerar o papel que as variáveis desempenham nelas—e, com base em algumas outras ideias, nomeadamente as de variáveis "livres" e "ligadas", podemos definir o que é uma sentença (nomeadamente, uma fórmula sem variáveis livres). Fazemos isso não apenas porque torna a definição de "sentença" mais gerenciável, mas também porque será crucial para a maneira como definimos a noção semântica de satisfação.

Vamos definir "fórmula" para uma linguagem simples de primeira ordem, contendo apenas um único predicado P e um único símbolo de constante a, e apenas os símbolos lógicos $\neg$, $\wedge$ e $\exists$. Nossas definições completas serão muito mais gerais: permitiremos infinitos predicados e símbolos de constante. De fato, também consideraremos símbolos de função que podem ser combinados com símbolos de constante e variáveis para formar "termos". Por enquanto, a e as variáveis serão nossos únicos termos. Precisamos de infinitas variáveis. Oficialmente usaremos os símbolos v_0, v_1, ..., como variáveis.

Definição 5.1. O conjunto de *fórmulas* Frm é definido da seguinte forma:

1. $P(a)$ e $P(v_i)$ são fórmulas ($i \in \mathbb{N}$).
2. Se A é uma fórmula, então $\neg A$ é uma fórmula.
3. Se A e B são fórmulas, então $(A \wedge B)$ é uma fórmula.
4. Se A é uma fórmula e x é uma variável, então $\exists x\, A$ é uma fórmula.
5. Nada mais é uma fórmula.

(1) nos diz que $P(a)$ e $P(v_i)$ são fórmulas, para qualquer $i \in \mathbb{N}$. Estas são as chamadas fórmulas *atômicas*. Elas nos dão algo com que começar. As outras cláusulas nos dão maneiras de formar novas fórmulas a partir daquelas que já formamos. Assim, por exemplo, por (2), obtemos que $\neg P(v_2)$ é uma fórmula, já que $P(v_2)$ já é uma fórmula por (1). Então, por (4), obtemos que $\exists v_2 \neg P(v_2)$ é outra fórmula, e assim por diante. (5) nos diz que *somente* cadeias que podemos formar desta maneira contam como fórmulas. Em particular, $\exists v_0 P(a)$ e $\exists v_0 \exists v_0 P(a)$ *contam* como fórmulas, e $(\neg P(a))$ não conta, por causa dos parênteses externos supérfluos.

Esta maneira de definir fórmulas é chamada de *definição indutiva*, e nos permite provar coisas sobre fórmulas usando uma versão de prova por indução chamada *indução estrutural*. Estas são discutidas de maneira geral em seção B.4 e seção B.5, que você deve revisar antes de mergulhar nas provas mais adiante. Basicamente, a ideia é que, se você quer dar uma prova de que algo é verdadeiro para todas as fórmulas, mostra primeiro que é verdadeiro para as fórmulas atômicas e então que *se* for verdadeiro para qualquer fórmula A (e B), *também* é verdadeiro para $\neg A$, $(A \wedge B)$ e $\exists x\, A$. Por exemplo, isso prova que é verdadeiro para $\exists v_2 \neg P(v_2)$: da primeira parte você sabe que é verdadeiro para a fórmula atômica $P(v_2)$. Então obtém que é verdadeiro para $\neg P(v_2)$ pela segunda parte, e novamente que é verdadeiro para $\exists v_2 \neg P(v_2)$ em si. Como todas as fórmulas são geradas indutivamente a partir de fórmulas atômicas, isso funciona para qualquer uma delas.

5.4 Satisfação

Já podemos avançar para a semântica da lógica de primeira ordem assim que sabemos o que são fórmulas: aqui, a definição básica é a de uma estrutura. Para nossa linguagem simples, uma estrutura $\boldsymbol{M}$ tem apenas três componentes: um conjunto não vazio $|\boldsymbol{M}|$ chamado de *domínio*, o que a designa em $\boldsymbol{M}$, e do que P é

verdadeiro em $\boldsymbol{M}$. O objeto designado por a é denotado por $a^{\boldsymbol{M}}$ e o conjunto de coisas das quais P é verdadeiro por $P^{\boldsymbol{M}}$. Uma estrutura $\boldsymbol{M}$ consiste apenas nestas três coisas: $|\boldsymbol{M}|$, $a^{\boldsymbol{M}} \in |\boldsymbol{M}|$ e $P^{\boldsymbol{M}} \subseteq |\boldsymbol{M}|$. O caso geral será mais complicado, pois haverá muitos predicados e símbolos de constante, os símbolos de constante podem ter mais de um lugar, e haverá também símbolos de função.

Isso basta para dar uma definição de satisfação para fórmulas que não contêm variáveis. A ideia é dar uma definição indutiva que espelha a maneira como definimos fórmulas. Especificamos quando uma fórmula atômica é satisfeita em $\boldsymbol{M}$, e então quando, por exemplo, $\neg A$ é satisfeita em $\boldsymbol{M}$ com base em se A é ou não satisfeita em $\boldsymbol{M}$. Por exemplo, poderíamos definir:

1. $P(a)$ é satisfeita em $\boldsymbol{M}$ se, e somente se, $a^{\boldsymbol{M}} \in P^{\boldsymbol{M}}$.

2. $\neg A$ é satisfeita em $\boldsymbol{M}$ se, e somente se, A não é satisfeita em $\boldsymbol{M}$.

3. $(A \wedge B)$ é satisfeita em $\boldsymbol{M}$ se, e somente se, A é satisfeita em $\boldsymbol{M}$ e B também é satisfeita em $\boldsymbol{M}$.

Digamos que $|\boldsymbol{M}| = \{0,1,2\}$, $a^{\boldsymbol{M}} = 1$ e $P^{\boldsymbol{M}} = \{1,2\}$. Esta definição nos diria que $P(a)$ é satisfeita em $\boldsymbol{M}$ (pois $a^{\boldsymbol{M}} = 1 \in \{1,2\} = P^{\boldsymbol{M}}$). Nos diria ainda que $\neg P(a)$ não é satisfeita em $\boldsymbol{M}$, e que, por sua vez, $\neg\neg P(a)$ é, e que $(\neg P(a) \wedge P(a))$ não é satisfeita, e assim por diante.

A dificuldade surge quando queremos dar uma definição para os quantificadores: gostaríamos de dizer algo como: "$\exists v_0\, P(v_0)$ é satisfeita se, e somente se, $P(v_0)$ é satisfeita." Mas a estrutura $\boldsymbol{M}$ não nos diz o que fazer com variáveis. O que realmente queremos dizer é que $P(v_0)$ é satisfeita *para algum valor de* v_0. Para tornar isso preciso, precisamos de uma maneira de atribuir elementos de $|\boldsymbol{M}|$ não apenas a a, mas também a v_0. Para isso, introduzimos *atribuições de variáveis*. Uma atribuição de variáveis é simplesmente uma função s que mapeia variáveis para elementos de $|\boldsymbol{M}|$ (em nosso exemplo, para um entre 1, 2 ou 3). Como não

sabemos de antemão quais variáveis podem aparecer em uma fórmula, não podemos limitar a quais variáveis s atribui valores. A solução simples é exigir que s atribua valores a *todas* as variáveis v_0, v_1, ... Usaremos apenas aquelas de que precisarmos.

Em vez de definir a satisfação de fórmulas apenas em relação a uma estrutura, definiremos em relação a uma estrutura $\boldsymbol{M}$ *e* uma atribuição de variáveis s, e escreveremos $\boldsymbol{M}, s \vDash A$ por brevidade. Nossa definição agora incluirá uma cláusula adicional para lidar com fórmulas atômicas contendo variáveis:

1. $\boldsymbol{M}, s \vDash P(a)$ se, e somente se, $a^{\boldsymbol{M}} \in P^{\boldsymbol{M}}$.
2. $\boldsymbol{M}, s \vDash P(v_i)$ se, e somente se, $s(v_i) \in P^{\boldsymbol{M}}$.
3. $\boldsymbol{M}, s \vDash \neg A$ se, e somente se, não $\boldsymbol{M}, s \vDash A$.
4. $\boldsymbol{M}, s \vDash (A \wedge B)$ se, e somente se, $\boldsymbol{M}, s \vDash A$ e $\boldsymbol{M}, s \vDash B$.

Ok, isso resolve um problema: agora podemos dizer quando $\boldsymbol{M}$ satisfaz $P(v_0)$ para o valor $s(v_0)$. Para obter a definição correta para $\exists v_0\, P(v_0)$ temos que fazer mais uma coisa: queremos que $\boldsymbol{M}, s \vDash \exists v_0\, P(v_0)$ se, e somente se, $\boldsymbol{M}, s' \vDash P(v_0)$ para *alguma* maneira s' de atribuir um valor a v_0. Mas o valor atribuído a v_0 não precisa necessariamente ser o valor que $s(v_0)$ escolhe. Introduziremos uma notação para isso: se $m \in |\boldsymbol{M}|$, então deixamos $s[m/v_0]$ ser a atribuição que é igual a s (para todas as variáveis diferentes de v_0), exceto que a v_0 atribui m. Agora nossa definição pode ser:

5. $\boldsymbol{M}, s \vDash \exists v_i\, A$ se, e somente se, $\boldsymbol{M}, s[m/v_i] \vDash A$ para algum $m \in |\boldsymbol{M}|$.

Funciona? Digamos que $s(v_i) = 0$ para todo $i \in \mathbb{N}$. $\boldsymbol{M}, s \vDash \exists v_0\, P(v_0)$ se, e somente se, há um $m \in |\boldsymbol{M}|$ tal que $\boldsymbol{M}, s[m/v_0] \vDash P(v_0)$. E há: podemos escolher $m = 1$ ou $m = 2$. Note que isso é verdadeiro mesmo que o valor $s(v_0)$ atribuído a v_0 por s—neste caso, 0—não faça o trabalho. Temos $\boldsymbol{M}, s[1/v_0] \vDash P(v_0)$, mas não $\boldsymbol{M}, s \vDash P(v_0)$.

Se isso parece confuso e trabalhoso: é. Mas a complexidade adicional é necessária para dar uma definição precisa e indutiva de satisfação para todas as fórmulas, e precisamos de algo assim para definir precisamente as noções semânticas. Há outras maneiras de fazer isso, mas todas são igualmente (in)elegantes.

5.5 Sentenças

Ok, agora temos uma (esboço de uma) definição de satisfação ("verdadeira em") para estruturas e fórmulas. Mas isso requer um ingrediente adicional—uma atribuição de variável—e o que queríamos era uma definição de sentenças. Como nos livramos das atribuições, e o que são sentenças?

Você provavelmente se lembra de uma discussão, em sua primeira introdução à lógica formal, sobre a relação entre variáveis e quantificadores. Um quantificador é sempre seguido por uma variável e, então, na parte da sentença à qual esse quantificador se aplica (seu "escopo"), entendemos que a variável está "ligada" a esse quantificador. Em fórmulas, não se exigia que toda variável tivesse um quantificador correspondente, e as variáveis sem quantificadores correspondentes são "livres" ou "não ligadas". Tomaremos as sentenças como sendo todas aquelas fórmulas que não têm variáveis livres.

Novamente, a ideia intuitiva de quando uma ocorrência de uma variável em uma fórmula A está ligada, de qual quantificador a liga e de quando ela é livre não é difícil de captar. Você pode ter aprendido um método para testar isso, talvez envolvendo a contagem de parênteses. Temos que insistir em uma definição precisa—e, como definimos fórmulas por indução, podemos dar uma definição das ocorrências livres e ligadas de uma variável x em uma fórmula A também por indução. Por exemplo, ela poderia ter este aspecto para nossa linguagem simplificada:

1. Se A é atômica, todas as ocorrências de x nela são livres (isto é, a ocorrência de x em $P(x)$ é livre).

2. Se A é da forma $\neg B$, então uma ocorrência de x em $\neg B$ é livre se, e somente se, a ocorrência correspondente de x for livre em B (isto é, as ocorrências livres de variáveis em B são exatamente as ocorrências correspondentes em $\neg B$).

3. Se A é da forma $(B \wedge C)$, então uma ocorrência de x em $(B \wedge C)$ é livre se, e somente se, a ocorrência correspondente de x for livre em B ou em C.

4. Se A é da forma $\exists x\, B$, então nenhuma ocorrência de x em A é livre; se é da forma $\exists y\, B$, em que y é uma variável diferente de x, então uma ocorrência de x em $\exists y\, B$ é livre se, e somente se, a ocorrência correspondente de x for livre em B.

Uma vez que tenhamos uma definição precisa de ocorrências livres e ligadas de variáveis, podemos simplesmente dizer: uma sentença é qualquer fórmula sem ocorrências livres de variáveis.

5.6 Noções Semânticas

Mencionamos acima que, quando consideramos se $M, s \vDash A$ vale, (por conveniência) deixamos s atribuir valores a todas as variáveis, mas apenas os valores que atribui às variáveis em A são usados. De fato, são apenas os valores das variáveis *livres* em A que importam. É claro que, por sermos cuidadosos, vamos provar esse fato. Como as sentenças não têm variáveis livres, s não importa de modo algum quando se trata de saber se são satisfeitas em uma estrutura. Assim, quando A é uma sentença, podemos definir $M \vDash A$ como significando "$M, s \vDash A$ para todo s," o que, aliás, é verdadeiro se, e somente se, $M, s \vDash A$ para pelo menos um s. Precisamos introduzir atribuições de variáveis para obter uma definição funcional de satisfação para fórmulas, mas para sentenças, a satisfação é independente das atribuições de variáveis.

Uma vez que temos uma definição de "$M \vDash A$," sabemos o que "caso" e "verdadeira em" significam no que diz respeito às sentenças da lógica de primeira ordem. Com base na definição de $M \vDash A$ para sentenças, podemos então definir as noções semânticas básicas de validade, consequência e satisfatibilidade. Uma sentença é válida, $\vDash A$, se toda estrutura a satisfaz. Ela é consequência de um conjunto de sentenças, $\Gamma \vDash A$, se toda estrutura que satisfaz todas as sentenças em Γ também satisfaz A. E um conjunto de sentenças é satisfatível se alguma estrutura satisfaz todas as sentenças nele ao mesmo tempo.

Como as fórmulas são definidas indutivamente, e a satisfação é, por sua vez, definida por indução na estrutura das fórmulas, podemos usar indução para provar propriedades da nossa semântica e relacionar as noções semânticas definidas. Vamos reunir e provar algumas dessas propriedades, em parte porque são individualmente interessantes, mas principalmente porque muitas delas serão úteis quando formos investigar a relação entre semântica e sistemas de derivação. Para isso, também teremos que definir (precisamente, isto é, por indução) algumas noções sintáticas e operações que ainda não mencionamos.

5.7 Substituição

Discutiremos um exemplo para ilustrar como as coisas se articulam e como o desenvolvimento da sintaxe e da semântica estabelece as bases para nossas investigações mais avançadas posteriormente. Nossos sistemas de derivação deveriam permitir-nos derivar $P(a)$ de $\forall v_0\, P(v_0)$. Talvez até queiramos afirmar isso como uma regra de inferência. No entanto, para fazê-lo, devemos ser capazes de enunciá-lo nos termos mais gerais: não apenas para P, a e v_0, mas para qualquer fórmula A, e termo t, e variável x. (Lembre-se de que símbolos de constante são termos, mas vamos considerar também termos mais complicados construídos a partir de símbolos de constante e símbolos de função.) Então, queremos ser capazes de dizer algo como, "sempre que você tiver

derivado $\forall x\, A(x)$ você está justificado ao inferir $A(t)$—o resultado da remoção de $\forall x$ e substituindo x por t." Mas o que exatamente significa "substituir x por t"? Qual é a relação entre $A(x)$ e $A(t)$? Isso sempre funciona?

Para tornar isso preciso, definimos a operação de *substituição*. A substituição é realmente complicada, porque não podemos simplesmente substituir todos os x em A por t, e nem todos os t podem ser substituídos por qualquer x. Lidaremos com isso novamente usando definições indutivas. Mas uma vez feito isso, especificando uma regra de inferência como "inferir $A(t)$ de $\forall x\, A(x)$" torna-se uma definição precisa. Além disso, seremos capazes de mostrar que esta é uma boa regra de inferência no sentido de que $A(t)$ é consequência de $\forall x\, A(x)$. Mas para provar isso, temos de provar novamente algo que à primeira vista pode levá-lo a perguntar "por que estamos fazendo isso?" Que $A(t)$ seja consequência de $\forall x\, A(x)$ depende do fato de que se $\boldsymbol{M} \vDash A(t)$ vale ou não depende apenas do valor do termo t, ou seja, se deixarmos m ser qualquer elemento de $|\boldsymbol{M}|$ que for selecionado por t, então $\boldsymbol{M}, s \vDash A(t)$ se, e somente se, $\boldsymbol{M}, s[m/x] \vDash A(x)$. Isso vale mesmo quando t contém variáveis, mas teremos que ter cuidado com a forma exata como declaramos o resultado.

5.8 Modelos e Teorias

Uma vez que definimos a sintaxe e a semântica da lógica de primeira ordem, podemos trabalhar investigando as propriedades de estruturas e as noções semânticas. Também podemos definir sistemas de derivação e investigá-los. Para um conjunto de sentenças, podemos perguntar: quais estruturas tornam todas as sentenças desse conjunto verdadeiras? Dado um conjunto de sentenças Γ, uma estrutura $\boldsymbol{M}$ que as satisfaz é chamada de *modelo de* Γ. Podemos começar de Γ e tentar encontrar seus modelos—como eles são? Quão grandes ou pequenos precisam ser? Mas também podemos começar com uma única estrutura ou coleção de estruturas e perguntar: quais sentenças são verdadeiras ne-

las? Há sentenças que *caracterizam* essas estruturas no sentido de que elas, e somente elas, são verdadeiras nelas? Esses tipos de questões são o domínio da *teoria dos modelos*. Elas também fundamentam o *método axiomático*: descrever uma coleção de estruturas por um conjunto de sentenças, os axiomas de uma teoria. Isso é possível pela observação de que exatamente aquelas sentenças que são consequência na lógica de primeira ordem dos axiomas são verdadeiras em todos os modelos dos axiomas.

Como exemplo muito simples, considere pré-ordens. Uma pré-ordem é uma relação R sobre algum conjunto A que é reflexiva e transitiva. Um conjunto A com uma relação binária $R \subseteq A \times A$ sobre ele é exatamente o que precisaríamos para dar uma estrutura para uma linguagem de primeira ordem com um único símbolo de predicado de dois lugares P: definiríamos $|\mathcal{M}| = A$ e $P^{\mathcal{M}} = R$. Como R é uma pré-ordem, é reflexiva e transitiva, e podemos encontrar um conjunto Γ de sentenças da lógica de primeira ordem que diz isso:

$$\forall v_0 \, P(v_0, v_0)$$
$$\forall v_0 \, \forall v_1 \, \forall v_2 \, ((P(v_0, v_1) \wedge P(v_1, v_2)) \rightarrow P(v_0, v_2))$$

Estas sentenças são apenas as simbolizações de "para todo x, Rxx" (R é reflexiva) e "sempre que Rxy e Ryz, então também Rxz" (R é transitiva). Vemos que uma estrutura $\mathcal{M}$ é um modelo dessas duas sentenças Γ se, e somente se, R (isto é, $P^{\mathcal{M}}$), é uma pré-ordem sobre A (isto é, $|\mathcal{M}|$). Em outras palavras, os modelos de Γ são exatamente as pré-ordens. Qualquer propriedade de todas as pré-ordens que possa ser expressa na linguagem de primeira ordem apenas com P como predicado (como reflexividade e transitividade acima), é consequência das duas sentenças em Γ e vice-versa. Assim, qualquer coisa que possamos provar sobre modelos de Γ provamos sobre todas as pré-ordens.

Para qualquer teoria particular e classe de modelos (como Γ e todas as pré-ordens), haverá questões interessantes sobre o que pode ser expresso na linguagem de primeira ordem correspondente, e o que não pode ser expresso. Há algumas propriedades

de estruturas que são interessantes para todas as linguagens e classes de modelos, a saber, aquelas que dizem respeito ao tamanho do domínio. Pode-se sempre expressar, por exemplo, que o domínio contém exatamente n elementos, para qualquer $n \in \mathbb{Z}^+$. Pode-se também expressar, usando um conjunto de infinitas sentenças, que o domínio é infinito. Mas não se pode expressar que o domínio é finito, ou que o domínio é incontável. Estes resultados sobre as limitações das linguagens de primeira ordem são consequências dos teoremas da compacidade e de Löwenheim–Skolem.

5.9 Correção e Completude

Também introduziremos sistemas de derivação para a lógica de primeira ordem. Há muitos sistemas de derivação que os lógicos desenvolveram, mas todos definem a mesma relação de derivabilidade entre sentenças. Dizemos que Γ *deriva* A, $\Gamma \vdash A$, se há uma derivação de um certo tipo precisamente definido. As derivações são sempre disposições finitas de símbolos—talvez uma lista de sentenças, ou alguma estrutura mais complicada. O propósito dos sistemas de derivação é fornecer uma ferramenta para determinar se uma sentença é consequência de algum conjunto Γ. Para servir a esse propósito, deve ser verdade que $\Gamma \vDash A$ se, e somente se, $\Gamma \vdash A$.

Se $\Gamma \vdash A$ mas não $\Gamma \vDash A$, nosso sistema de derivação seria forte demais, provaria demais. A propriedade de que se $\Gamma \vdash A$ então $\Gamma \vDash A$ é chamada de *correção*, e é um requisito mínimo para qualquer bom sistema de derivação. Por outro lado, se $\Gamma \vDash A$ mas não $\Gamma \vdash A$, então nosso sistema de derivação é fraco demais, não prova o suficiente. A propriedade de que se $\Gamma \vDash A$ então $\Gamma \vdash A$ é chamada de *completude*. A correção é usualmente relativamente fácil de provar (por indução na estrutura das derivações, que são definidas indutivamente). A completude é mais difícil de provar.

Correção e completude têm várias consequências importantes. Se um conjunto de sentenças Γ deriva uma contradição

(como $A \wedge \neg A$), ele é chamado de *inconsistente*. Γs inconsistentes não podem ter modelos; são insatisfatíveis. Da completude segue a contrapositiva: qualquer Γ que não seja inconsistente—ou, como diremos, *consistente*—tem um modelo. De fato, isso é equivalente à completude, e é a forma de completude que de fato provaremos. É um resultado profundo e talvez surpreendente: apenas o fato de que você não pode provar $A \wedge \neg A$ a partir de Γ garante que há uma estrutura que é como Γ a descreve. Assim, a completude dá uma resposta à pergunta: quais conjuntos de sentenças têm modelos? Resposta: todos e somente os conjuntos consistentes têm.

Os teoremas de correção e completude têm duas consequências importantes: a compacidade e o teorema de Löwenheim–Skolem. Estes são resultados importantes na teoria de modelos e podem ser usados para estabelecer muitos resultados interessantes. Já mencionamos dois: a lógica de primeira ordem não pode expressar que o domínio de uma estrutura é finito ou que é incontável.

Historicamente, tudo isso—como definir a sintaxe e a semântica da lógica de primeira ordem, como definir bons sistemas de derivação, como provar que são corretos e completos, ficar claro sobre o que pode e não pode ser expresso em linguagens de primeira ordem—levou muito tempo para ser descoberto e acertado. Agora sabemos como fazer isso, mas percorrer todos os detalhes ainda pode ser confuso e tedioso. Mas também é importante, porque os métodos desenvolvidos aqui para a linguagem formal da lógica de primeira ordem são aplicados em todo lugar em lógica, ciência da computação e linguística. Assim, trabalhar os detalhes compensa a longo prazo.

CAPÍTULO 6

Sintaxe da Lógica de Primeira Ordem

6.1 Introdução

Para desenvolver a teoria e a metateoria da lógica de primeira ordem, devemos primeiro definir a sintaxe e a semântica de suas expressões. As expressões da lógica de primeira ordem são termos e fórmulas. Termos são formados a partir de variáveis, símbolos de constante e símbolos de função. Fórmulas, por sua vez, são formadas a partir de predicados juntamente com termos (estes formam as menores fórmulas, as "atômicas"), e então, a partir de fórmulas atômicas, podemos formar outras mais complexas usando conectivos lógicos e quantificadores. Há muitas maneiras diferentes de estabelecer as regras de formação; damos apenas uma possível. Outros sistemas escolherão símbolos diferentes, selecionarão conjuntos diferentes de conectivos como primitivos,

usarão parênteses de forma diferente (ou nem mesmo os usarão, como no caso da chamada notação polonesa). O que todas as abordagens têm em comum, no entanto, é que as regras de formação definem o conjunto de termos e fórmulas *indutivamente*. Se feito corretamente, cada expressão pode resultar essencialmente de uma única maneira de acordo com as regras de formação. A definição indutiva que resulta em expressões que são *de leitura única* significa que podemos dar significados a essas expressões usando o mesmo método—a definição indutiva.

6.2 Linguagens de Primeira Ordem

As expressões da lógica de primeira ordem são construídas a partir de um vocabulário básico contendo *variáveis*, *símbolos de constante*, *predicados* e às vezes *símbolos de função*. A partir deles, juntamente com conectivos lógicos, quantificadores e símbolos de pontuação como parênteses e vírgulas, formam-se *termos* e *fórmulas*.

Informalmente, predicados são nomes para propriedades e relações, símbolos de constante são nomes para objetos individuais, e símbolos de função são nomes para mapeamentos. Estes, exceto a identidade $=$, são os *símbolos não lógicos* e juntos constituem uma linguagem. Qualquer linguagem de primeira ordem $\mathscr{L}$ é determinada por seus símbolos não lógicos. No caso mais geral, $\mathscr{L}$ contém infinitos símbolos de cada tipo.

No caso geral, fazemos uso dos seguintes símbolos na lógica de primeira ordem:

1. Símbolos lógicos

    a) Conectivos lógicos: $\neg$ (negação), $\wedge$ (conjunção), $\vee$ (disjunção), $\rightarrow$ (condicional), $\forall$ (quantificador universal), $\exists$ (quantificador existencial).

    b) A constante proposicional para falsidade $\bot$.

    c) A identidade de dois lugares $=$.

d) Um conjunto infinito contável de variáveis: v_0, v_1, v_2, ...

2. Símbolos não lógicos, que constituem a *linguagem padrão* da lógica de primeira ordem

   a) Um conjunto infinito contável de predicados de n lugares para cada $n > 0$: A^n_0, A^n_1, A^n_2, ...
   b) Um conjunto infinito contável de símbolos de constante: c_0, c_1, c_2, ...
   c) Um conjunto infinito contável de símbolos de função de n lugares para cada $n > 0$: f^n_0, f^n_1, f^n_2, ...

3. Sinais de pontuação: (,) e a vírgula.

A maioria de nossas definições e resultados será formulada para a linguagem padrão completa da lógica de primeira ordem. No entanto, dependendo da aplicação, também podemos restringir a linguagem a apenas alguns predicados, símbolos de constante e símbolos de função.

Exemplo 6.1. A linguagem $\mathscr{L}_A$ da aritmética contém um único predicado de dois lugares $<$, um único símbolo de constante 0, um símbolo de função de um lugar $'$, e dois símbolos de função de dois lugares $+$ e $\times$.

Exemplo 6.2. A linguagem da teoria dos conjuntos $\mathscr{L}_Z$ contém apenas o único predicado de dois lugares $\in$.

Exemplo 6.3. A linguagem das ordens $\mathscr{L}_\leq$ contém apenas o predicado de dois lugares $\leq$.

Novamente, estas são convenções: oficialmente, estes são apenas apelidos, por exemplo, $<$, $\in$ e $\leq$ são apelidos para A^2_0, 0 para c_0, $'$ para f^1_0, $+$ para f^2_0, $\times$ para f^2_1.

Além dos conectivos primitivos e quantificadores introduzidos acima, também usamos os seguintes símbolos *definidos*: $\leftrightarrow$ (bicondicional), verdade $\top$

Um símbolo definido não faz parte oficialmente da linguagem, mas é introduzido como uma abreviação informal: ele nos permite abreviar fórmulas que, se usássemos apenas símbolos primitivos, ficariam bastante longas. Isso é obviamente uma vantagem. A maior vantagem, no entanto, é que as provas ficam mais curtas. Se um símbolo é primitivo, ele tem que ser tratado separadamente nas provas. Quanto mais símbolos primitivos, portanto, mais longas nossas provas.

Você pode estar familiarizado com terminologia e símbolos diferentes dos que usamos acima. Textos de lógica (e professores) comumente usam $\sim$, $\neg$, ou ! para "negação", $\wedge$, $\cdot$, ou & para "conjunção". Símbolos comumente usados para o "condicional" ou "implicação" são $\rightarrow$, $\Rightarrow$, e $\supset$. Símbolos para "bicondicional", "bi-implicação", ou "equivalência (material)" são $\leftrightarrow$, $\Leftrightarrow$, e $\equiv$. O símbolo $\bot$ é chamado de várias formas: "falsidade", "falsum", "absurdo", ou "bottom". O símbolo $\top$ é chamado de várias formas: "verdade", "verum", ou "top".

É convencional usar letras minúsculas (por exemplo, a, b, c) do início do alfabeto latino para símbolos de constante (às vezes chamados de nomes), e letras minúsculas do final (por exemplo, x, y, z) para variáveis. Quantificadores combinam-se com variáveis, por exemplo, x; variações notacionais incluem $\forall x$, $(\forall x)$, (x), Πx, $\bigwedge_x$ para o quantificador universal e $\exists x$, $(\exists x)$, (Ex), Σx, $\bigvee_x$ para o quantificador existencial.

Poderíamos tratar todos os operadores proposicionais e ambos os quantificadores como símbolos primitivos da linguagem. Poderíamos, em vez disso, escolher um estoque menor de símbolos primitivos e tratar os outros operadores como definidos. Conjuntos "funcionalmente completos para a verdade" de operadores booleanos incluem $\{\neg, \vee\}$, $\{\neg, \wedge\}$, e $\{\neg, \rightarrow\}$—estes podem ser combinados com qualquer um dos quantificadores para uma linguagem de primeira ordem expressivamente completa.

Você pode estar familiarizado com dois outros operadores: a barra de Sheffer $|$ (em homenagem a Henry Sheffer), e a seta de Peirce $\downarrow$, também conhecida como adaga de Quine. Quando recebem suas leituras usuais de "nand" e "nor" (respectivamente),

esses operadores são funcionalmente completos para a verdade por si sós.

6.3 Termos e Fórmulas

Uma vez dada uma linguagem de primeira ordem $\mathscr{L}$, podemos definir expressões construídas a partir do vocabulário básico de $\mathscr{L}$. Estas incluem, em particular, *termos* e *fórmulas*.

Definição 6.4 (Termos). O conjunto de *termos* $\mathrm{Trm}(\mathscr{L})$ de $\mathscr{L}$ é definido indutivamente por:

1. Toda variável é um termo.
2. Todo símbolo de constante de $\mathscr{L}$ é um termo.
3. Se f é um símbolo de função de n lugares e $t_1, \ldots, t_n$ são termos, então $f(t_1, \ldots, t_n)$ é um termo.
4. Nada mais é um termo.

Um termo que não contém variáveis é um *termo fechado*.

Os símbolos de constante aparecem em nossa especificação da linguagem e dos termos como uma categoria separada de símbolos, mas eles poderiam, em vez disso, ter sido incluídos como símbolos de função de zero lugares. Poderíamos então dispensar a segunda cláusula da definição de termos. Bastaria entender $f(t_1, \ldots, t_n)$ como apenas f sozinho se $n = 0$.

Definição 6.5 (Fórmulas). O conjunto de *fórmulas* $\mathrm{Frm}(\mathscr{L})$ da linguagem $\mathscr{L}$ é definido indutivamente como segue:

1. $\bot$ é uma fórmula atômica.
2. Se R é um predicado de n lugares de $\mathscr{L}$ e $t_1, \ldots, t_n$ são termos de $\mathscr{L}$, então $R(t_1, \ldots, t_n)$ é uma fórmula atômica.

3. Se t_1 e t_2 são termos de $\mathscr{L}$, então $=(t_1, t_2)$ é uma fórmula atômica.

4. Se A é uma fórmula, então $\neg A$ é uma fórmula.

5. Se A e B são fórmulas, então $(A \wedge B)$ é uma fórmula.

6. Se A e B são fórmulas, então $(A \vee B)$ é uma fórmula.

7. Se A e B são fórmulas, então $(A \rightarrow B)$ é uma fórmula.

8. Se A é uma fórmula e x é uma variável, então $\forall x\, A$ é uma fórmula.

9. Se A é uma fórmula e x é uma variável, então $\exists x\, A$ é uma fórmula.

10. Nada mais é uma fórmula.

As definições do conjunto de termos e do conjunto de fórmulas são *definições indutivas*. Essencialmente, construímos o conjunto de fórmulas em infinitos estágios. No estágio inicial, declaramos que todas as fórmulas atômicas são fórmulas; isso corresponde aos primeiros casos da definição, isto é, os casos para $\bot$, $R(t_1, \ldots, t_n)$ e $=(t_1, t_2)$. "Fórmula atômica" significa, portanto, qualquer fórmula desta forma.

Os outros casos da definição dão regras para construir novas fórmulas a partir de fórmulas já construídas. No segundo estágio, podemos usá-los para construir fórmulas a partir de fórmulas atômicas. No terceiro estágio, construímos novas fórmulas a partir das fórmulas atômicas e daquelas obtidas no segundo estágio, e assim por diante. Uma fórmula é qualquer coisa que seja eventualmente construída em tal estágio, e nada mais.

Por convenção, escrevemos $=$ entre seus argumentos e omitimos os parênteses: $t_1 = t_2$ é uma abreviação para $=(t_1, t_2)$. Além disso, $\neg=(t_1, t_2)$ é abreviado como $t_1 \neq t_2$. Ao escrever uma fórmula $(B * C)$ construída a partir de B, C usando um conectivo

de dois lugares $*$, frequentemente omitiremos o par de parênteses mais externo e escreveremos simplesmente $B * C$.

Alguns textos de lógica exigem que a variável x deva ocorrer em A para que $\exists x\, A$ e $\forall x\, A$ contem como fórmulas. Nada de ruim acontece se você não exigir isso, e isso torna as coisas mais fáceis.

Definição 6.6. As fórmulas construídas usando os operadores definidos devem ser entendidas como segue:

1. $\top$ abrevia $\neg\bot$.
2. $A \leftrightarrow B$ abrevia $(A \to B) \wedge (B \to A)$.

Se trabalhamos em uma linguagem para uma aplicação específica, frequentemente escreveremos predicados e símbolos de função de dois lugares entre os respectivos termos, por exemplo, $t_1 < t_2$ e $(t_1 + t_2)$ na linguagem da aritmética e $t_1 \in t_2$ na linguagem da teoria dos conjuntos. A função sucessora na linguagem da aritmética é até escrita convencionalmente *depois* de seu argumento: t'. Oficialmente, no entanto, estas são apenas abreviações convencionais para $A^2_0(t_1, t_2)$, $f^2_0(t_1, t_2)$, $A^2_0(t_1, t_2)$ e $f^1_0(t)$, respectivamente.

Definição 6.7 (Identidade sintática). O símbolo $\equiv$ expressa a identidade sintática entre cadeias de símbolos, isto é, $A \equiv B$ sse A e B são cadeias de símbolos do mesmo comprimento e que contêm o mesmo símbolo em cada posição.

O símbolo $\equiv$ pode ser ladeado por cadeias obtidas por concatenação, por exemplo, $A \equiv (B \vee C)$ significa: a cadeia de símbolos A é a mesma cadeia que aquela obtida concatenando um parêntese de abertura, a cadeia B, o símbolo $\vee$, a cadeia C, e um parêntese de fechamento, nesta ordem. Se este é o caso, então sabemos que o primeiro símbolo de A é um parêntese de abertura, A contém B como uma subcadeia (começando no segundo símbolo), essa subcadeia é seguida por $\vee$, etc.

Como termos e fórmulas são construídos a partir de elementos básicos via definições indutivas, podemos usar os seguintes princípios de indução para provar coisas sobre eles.

Lema 6.8 (*Princípio de indução sobre termos*). *Seja $\mathcal{L}$ uma linguagem de primeira ordem. Se uma propriedade P é tal que*

1. *ela vale para toda variável v,*
2. *ela vale para todo símbolo de constante a de $\mathcal{L}$, e*
3. *ela vale para $f(t_1, \ldots, t_n)$ sempre que vale para $t_1, \ldots, t_n$ e f é um símbolo de função de n lugares de $\mathcal{L}$*

(supondo que $t_1, \ldots, t_n$ são termos de $\mathcal{L}$), então P vale para todo termo em $\mathrm{Trm}(\mathcal{L})$.

Lema 6.9 (*Princípio de indução sobre fórmulas*). *Seja $\mathcal{L}$ uma linguagem de primeira ordem. Se uma propriedade P vale para todas as fórmulas atômicas e é tal que*

1. *ela vale para $\neg A$ sempre que vale para A;*
2. *ela vale para $(A \wedge B)$ sempre que vale para A e B;*
3. *ela vale para $(A \vee B)$ sempre que vale para A e B;*
4. *ela vale para $(A \to B)$ sempre que vale para A e B;*
5. *ela vale para $\exists x\, A$ sempre que vale para A;*
6. *ela vale para $\forall x\, A$ sempre que vale para A;*

(supondo que A e B são fórmulas de $\mathcal{L}$), então P vale para todas as fórmulas em $\mathrm{Frm}(\mathcal{L})$.

6.4 Leitura Única

A maneira como definimos fórmulas garante que toda fórmula tem uma *leitura única*, isto é, há essencialmente apenas uma maneira de construí-la de acordo com nossas regras de formação para fórmulas e apenas uma maneira de "interpretá-la". Se não fosse assim, teríamos fórmulas ambíguas, isto é, fórmulas que têm mais de uma leitura ou interpretação—e isso é claramente algo que queremos evitar. Mas, mais importante, sem essa propriedade, a maioria das definições e provas que vamos dar não funcionaria.

Talvez a melhor maneira de deixar isso claro seja ver o que aconteceria se tivéssemos dado regras ruins para formar fórmulas que não garantissem a leitura única. Por exemplo, poderíamos ter esquecido os parênteses nas regras de formação para conectivos, por exemplo, poderíamos ter permitido isto:

> Se A e B são fórmulas, então $A \to B$ também é.

Partindo de uma fórmula atômica D, isso nos permitiria formar $D \to D$. A partir disso, juntamente com D, obteríamos $D \to D \to D$. Mas há duas maneiras de fazer isso:

1. Tomamos D como A e $D \to D$ como B.

2. Tomamos A como $D \to D$ e B é D.

Correspondentemente, há duas maneiras de "ler" a fórmula $D \to D \to D$. Ela é da forma $B \to C$ onde B é D e C é $D \to D$, mas *ela é também* da forma $B \to C$ com B sendo $D \to D$ e C sendo D.

Se isso acontece, nossas definições nem sempre funcionarão. Por exemplo, quando definimos o operador principal de uma fórmula, dizemos: em uma fórmula da forma $B \to C$, o operador principal é a ocorrência indicada de $\to$. Mas se podemos casar a fórmula $D \to D \to D$ com $B \to C$ das duas maneiras diferentes mencionadas acima, então em um caso obtemos a primeira ocorrência de $\to$ como o operador principal, e no segundo caso

a segunda ocorrência. Mas pretendemos que o operador principal seja uma *função* da fórmula, isto é, toda fórmula deve ter exatamente uma ocorrência de operador principal.

Lema 6.10. *O número de parênteses esquerdos e direitos em uma fórmula A é igual.*

Prova. Provamos isso por indução sobre a maneira como A é construída. Isso requer duas coisas: (a) Temos primeiro que provar que todas as fórmulas atômicas têm a propriedade em questão (a base da indução). (b) Então temos que provar que, quando construímos novas fórmulas a partir de fórmulas dadas, as novas fórmulas têm a propriedade, desde que as antigas a tenham.

Seja $l(A)$ o número de parênteses esquerdos, e $r(A)$ o número de parênteses direitos em A, e $l(t)$ e $r(t)$ similarmente o número de parênteses esquerdos e direitos em um termo t.

1. $A \equiv \bot$: A tem 0 parênteses esquerdos e 0 parênteses direitos.

2. $A \equiv R(t_1, \ldots, t_n)$: $l(A) = 1 + l(t_1) + \cdots + l(t_n) = 1 + r(t_1) + \cdots + r(t_n) = r(A)$. Aqui usamos o fato, deixado como exercício, de que $l(t) = r(t)$ para qualquer termo t.

3. $A \equiv t_1 = t_2$: $l(A) = l(t_1) + l(t_2) = r(t_1) + r(t_2) = r(A)$.

4. $A \equiv \neg B$: Por hipótese de indução, $l(B) = r(B)$. Assim $l(A) = l(B) = r(B) = r(A)$.

5. $A \equiv (B * C)$: Por hipótese de indução, $l(B) = r(B)$ e $l(C) = r(C)$. Assim $l(A) = 1 + l(B) + l(C) = 1 + r(B) + r(C) = r(A)$.

6. $A \equiv \forall x\, B$: Por hipótese de indução, $l(B) = r(B)$. Assim, $l(A) = l(B) = r(B) = r(A)$.

7. $A \equiv \exists x\, B$: Analogamente. □

Definição 6.11 (Prefixo próprio). Uma cadeia de símbolos B é um *prefixo próprio* de uma cadeia de símbolos A se a concatenação de B e uma cadeia de símbolos não vazia resulta em A.

Lema 6.12. *Se A é uma fórmula, e B é um prefixo próprio de A, então B não é uma fórmula.*

Prova. Exercício. □

Proposição 6.13. *Se A é uma fórmula atômica, então ela satisfaz uma, e apenas uma, das seguintes condições.*

1. $A \equiv \bot$.
2. $A \equiv R(t_1, \ldots, t_n)$ *onde R é um predicado de n lugares, $t_1, \ldots, t_n$ são termos, e cada um de R, $t_1, \ldots, t_n$ é determinado de forma única.*
3. $A \equiv t_1 = t_2$ *onde t_1 e t_2 são termos determinados de forma única.*

Prova. Exercício. □

Proposição 6.14 (Leitura Única). *Toda fórmula satisfaz uma, e apenas uma, das seguintes condições.*

1. *A é atômica.*
2. *A é da forma $\neg B$.*
3. *A é da forma $(B \wedge C)$.*
4. *A é da forma $(B \vee C)$.*
5. *A é da forma $(B \to C)$.*
6. *A é da forma $\forall x\, B$.*

7. *A é da forma $\exists x\, B$.*

Além disso, em cada caso B, ou B e C, são determinados de forma única. Isso significa que, por exemplo, não há pares diferentes B, C e B', C' tais que A seja ao mesmo tempo da forma $(B \to C)$ e $(B' \to C')$.

Prova. As regras de formação exigem que, se uma fórmula não é atômica, ela deve começar com um parêntese de abertura (, $\neg$, ou um quantificador. Por outro lado, toda fórmula que começa com um dos seguintes símbolos deve ser atômica: um predicado, um símbolo de função, um símbolo de constante, $\bot$.

Então, na verdade só temos que mostrar que se A é da forma $(B * C)$ e também da forma $(B' *' C')$, então $B \equiv B'$, $C \equiv C'$, e $* = *'$.

Então suponha que tanto $A \equiv (B * C)$ quanto $A \equiv (B' *' C')$. Então ou $B \equiv B'$ ou não. Se for, claramente $* = *'$ e $C \equiv C'$, pois elas então são subcadeias de A que começam no mesmo lugar e têm o mesmo comprimento. O outro caso é $B \not\equiv B'$. Como B e B' são ambas subcadeias de A que começam no mesmo lugar, uma deve ser um prefixo próprio da outra. Mas isso é impossível por Lema 6.12. □

6.5 Operador principal de uma Fórmula

Frequentemente é útil falar sobre o último operador usado na construção de uma fórmula A. Esse operador é chamado de *operador principal* de A. Intuitivamente, é o operador "mais externo" de A. Por exemplo, o operador principal de $\neg A$ é $\neg$, o operador principal de $(A \vee B)$ é $\vee$, etc.

Definição 6.15 (Operador principal). O *operador principal* de uma fórmula A é definido como segue:

1. A é atômica: A não tem operador principal.

2. $A \equiv \neg B$: o operador principal de A é $\neg$.
3. $A \equiv (B \wedge C)$: o operador principal de A é $\wedge$.
4. $A \equiv (B \vee C)$: o operador principal de A é $\vee$.
5. $A \equiv (B \to C)$: o operador principal de A é $\to$.
6. $A \equiv \forall x\, B$: o operador principal de A é $\forall$.
7. $A \equiv \exists x\, B$: o operador principal de A é $\exists$.

Em cada caso, pretendemos a *ocorrência* específica indicada do operador principal na fórmula. Por exemplo, como a fórmula $((D \to E) \to (E \to D))$ é da forma $(B \to C)$ onde B é $(D \to E)$ e C é $(E \to D)$, a segunda ocorrência de $\to$ é o operador principal.

Esta é uma definição *recursiva* de uma função que mapeia todas as fórmulas não atômicas para sua ocorrência de operador principal. Por causa da maneira como as fórmulas são definidas indutivamente, toda fórmula A satisfaz um dos casos em Definição 6.15. Isso garante que, para cada fórmula A não atômica, existe um operador principal. Como cada fórmula satisfaz apenas uma dessas condições, e como as fórmulas menores a partir das quais A é construída são determinadas de forma única em cada caso, a ocorrência de operador principal de A é única, e assim definimos uma função.

Chamamos as fórmulas pelos nomes em Tabela 6.1 dependendo de qual símbolo é seu operador principal. Lembre-se, no entanto, de que os operadores definidos não aparecem oficialmente em fórmulas. Eles são apenas abreviações, então oficialmente não podem ser o operador principal de uma fórmula. Em provas sobre todas as fórmulas, eles, portanto, não precisam ser tratados separadamente.

Operador principal	Tipo de fórmula	Exemplo
nenhum	(fórmula) atômica	$\bot$, $R(t_1,\ldots,t_n)$, $t_1 = t_2$
$\neg$	negação	$\neg A$
$\wedge$	conjunção	$(A \wedge B)$
$\vee$	disjunção	$(A \vee B)$
$\rightarrow$	condicional	$(A \rightarrow B)$
$\leftrightarrow$	bicondicional	$(A \leftrightarrow B)$
$\forall$	(fórmula) universal	$\forall x\, A$
$\exists$	(fórmula) existencial	$\exists x\, A$

Tabela 6.1: Operador principal e nomes de fórmulas

6.6 Subfórmulas

Frequentemente é útil falar sobre as fórmulas que "compõem" uma dada fórmula. Chamamos estas de suas *subfórmulas*. Qualquer fórmula conta como uma subfórmula de si mesma; uma subfórmula de A diferente do próprio A é uma *subfórmula própria*.

Definição 6.16 (Subfórmula imediata). Se A é uma fórmula, as *subfórmulas imediatas* de A são definidas indutivamente como segue:

1. Fórmulas atômicas não têm subfórmulas imediatas.
2. $A \equiv \neg B$: A única subfórmula imediata de A é B.
3. $A \equiv (B * C)$: As subfórmulas imediatas de A são B e C ($*$ é qualquer um dos conectivos de dois lugares).
4. $A \equiv \forall x\, B$: A única subfórmula imediata de A é B.
5. $A \equiv \exists x\, B$: A única subfórmula imediata de A é B.

Definição 6.17 (Subfórmula própria). Se A é uma fórmula, as *subfórmulas próprias* de A são definidas recursivamente como segue:

1. Fórmulas atômicas não têm subfórmulas próprias.
2. $A \equiv \neg B$: As subfórmulas próprias de A são B juntamente com todas as subfórmulas próprias de B.
3. $A \equiv (B * C)$: As subfórmulas próprias de A são B, C, juntamente com todas as subfórmulas próprias de B e as de C.
4. $A \equiv \forall x\, B$: As subfórmulas próprias de A são B juntamente com todas as subfórmulas próprias de B.
5. $A \equiv \exists x\, B$: As subfórmulas próprias de A são B juntamente com todas as subfórmulas próprias de B.

Definição 6.18 (Subfórmula). As subfórmulas de A são o próprio A juntamente com todas as suas subfórmulas próprias.

Note a diferença sutil em como definimos subfórmulas imediatas e subfórmulas próprias. No primeiro caso, definimos diretamente as subfórmulas imediatas de uma fórmula A para cada forma possível de A. É uma definição explícita por casos, e os casos espelham a definição indutiva do conjunto de fórmulas. No segundo caso, também espelhamos a maneira como o conjunto de todas as fórmulas é definido, mas em cada caso também incluímos as subfórmulas próprias das fórmulas menores B, C além dessas próprias fórmulas. Isso torna a definição *recursiva*. Em geral, uma definição de uma função sobre um conjunto definido indutivamente (no nosso caso, fórmulas) é recursiva se os casos na definição da função fazem uso da própria função. Para ser bem definida, devemos garantir, no entanto, que só usamos os valores da função para argumentos que vêm "antes" daquele que

estamos definindo—no nosso caso, ao definir "subfórmula própria" para $(B * C)$ usamos apenas as subfórmulas próprias das fórmulas "anteriores" B e C.

Proposição 6.19. *Suponha que B é uma subfórmula de A e C é uma subfórmula de B. Então C é uma subfórmula de A. Em outras palavras, a relação de subfórmula é transitiva.*

Proposição 6.20. *Suponha que A é uma fórmula com n conectivos e quantificadores. Então A tem no máximo $2n + 1$ subfórmulas.*

6.7 Sequências de Formação

Definir fórmulas via uma definição indutiva, e a técnica complementar de provar propriedades de fórmulas via indução, é uma abordagem elegante e eficiente. No entanto, também pode ser útil considerar uma abordagem mais ascendente, passo a passo, da construção de fórmulas, o que fazemos aqui usando a noção de uma *sequência de formação*. Para mostrar como termos e fórmulas podem ser introduzidos dessa maneira sem precisar referir-se a suas definições indutivas, primeiro introduzimos a noção de uma cadeia arbitrária de símbolos extraída de alguma linguagem $\mathscr{L}$.

Definição 6.21 (Cadeias). Suponha que $\mathscr{L}$ é uma linguagem de primeira ordem. Uma *$\mathscr{L}$-cadeia* é uma sequência finita de símbolos de $\mathscr{L}$. Quando a linguagem $\mathscr{L}$ está claramente fixada pelo contexto, frequentemente nos referiremos a uma $\mathscr{L}$-cadeia simplesmente como uma *cadeia*.

Exemplo 6.22. Para qualquer linguagem de primeira ordem $\mathscr{L}$, todas as $\mathscr{L}$-fórmulas são $\mathscr{L}$-cadeias, mas não reciprocamente. Por exemplo,

$$)(v_0 \rightarrow \exists$$

é uma $\mathscr{L}$-cadeia mas não uma $\mathscr{L}$-fórmula.

Definição 6.23 (Sequências de formação para termos). Uma sequência finita de $\mathscr{L}$-cadeias $\langle t_0, \ldots, t_n \rangle$ é uma *sequência de formação* para um termo t se $t \equiv t_n$ e para todo $i \leq n$, ou t_i é uma variável ou um símbolo de constante, ou $\mathscr{L}$ contém um símbolo de função f k-ário e existem $m_0, \ldots, m_k < i$ tais que $t_i \equiv f(t_{m_0}, \ldots, t_{m_k})$. Quando for necessário distinguir, nos referiremos às sequências de formação para termos como *sequências de formação de termos.*

Exemplo 6.24. A sequência

$$\langle c_0, v_0, f_0^2(c_0, v_0), f_0^1(f_0^2(c_0, v_0)) \rangle$$

é uma sequência de formação para o termo $f_0^1(f_0^2(c_0, v_0))$, assim como

$$\langle v_0, c_0, f_0^2(c_0, v_0), f_0^1(f_0^2(c_0, v_0)) \rangle.$$

Definição 6.25 (Sequências de formação para fórmulas). Uma sequência finita de $\mathscr{L}$-cadeias $\langle A_0, \ldots, A_n \rangle$ é uma *sequência de formação* para A se $A \equiv A_n$ e para todo $i \leq n$, ou A_i é uma fórmula atômica ou existem $j, k < i$ e uma variável x tais que uma das seguintes condições vale:

1. $A_i \equiv \neg A_j$.
2. $A_i \equiv (A_j \wedge A_k)$.
3. $A_i \equiv (A_j \vee A_k)$.
4. $A_i \equiv (A_j \rightarrow A_k)$.
5. $A_i \equiv \forall x\, A_j$.
6. $A_i \equiv \exists x\, A_j$.

Quando for necessário distinguir, nos referiremos às sequências de formação para fórmulas como *sequências de formação de fórmulas.*

Exemplo 6.26.

$$\langle A_0^1(v_0), A_1^1(c_1), (A_1^1(c_1) \wedge A_0^1(v_0)), \exists v_0\, (A_1^1(c_1) \wedge A_0^1(v_0))\rangle$$

é uma sequência de formação de $\exists v_0\, (A_1^1(c_1) \wedge A_0^1(v_0))$, assim como

$$\langle A_0^1(v_0), A_1^1(c_1), (A_1^1(c_1) \wedge A_0^1(v_0)), A_1^1(c_1), \\ \forall v_1\, A_0^1(v_0), \exists v_0\, (A_1^1(c_1) \wedge A_0^1(v_0))\rangle.$$

Como pode ser visto no segundo exemplo, sequências de formação podem conter "lixo": fórmulas que são redundantes ou não contribuem para a construção.

Proposição 6.27. *Toda fórmula A em* $\mathrm{Frm}(\mathscr{L})$ *tem uma sequência de formação.*

Prova. Suponha que A é atômica. Então a sequência $\langle A\rangle$ é uma sequência de formação para A. Agora suponha que B e C têm sequências de formação $\langle B_0, \ldots, B_n\rangle$ e $\langle C_0, \ldots, C_m\rangle$ respectivamente.

1. Se $A \equiv \neg B$, então $\langle B_0, \ldots, B_n, \neg B_n\rangle$ é uma sequência de formação para A.

2. Se $A \equiv (B \wedge C)$, então $\langle B_0, \ldots, B_n, C_0, \ldots, C_m, (B_n \wedge C_m)\rangle$ é uma sequência de formação para A.

3. Se $A \equiv (B \vee C)$, então $\langle B_0, \ldots, B_n, C_0, \ldots, C_m, (B_n \vee C_m)\rangle$ é uma sequência de formação para A.

4. Se $A \equiv (B \to C)$, então $\langle B_0, \ldots, B_n, C_0, \ldots, C_m, (B_n \to C_m)\rangle$ é uma sequência de formação para A.

5. Se $A \equiv \forall x\, B$, então $\langle B_0, \ldots, B_n, \forall x\, B_n \rangle$ é uma sequência de formação para A.

6. Se $A \equiv \exists x\, B$, então $\langle B_0, \ldots, B_n, \exists x\, B_n \rangle$ é uma sequência de formação para A.

Pelo princípio de indução sobre fórmulas, toda fórmula tem uma sequência de formação. □

Também podemos provar a recíproca. Isso é importante porque mostra que nossas duas maneiras de definir fórmulas são equivalentes: elas dão os mesmos resultados. Isso também significa que podemos provar teoremas sobre fórmulas usando indução ordinária sobre o comprimento das sequências de formação.

Lema 6.28. *Suponha que $\langle A_0, \ldots, A_n \rangle$ é uma sequência de formação para A_n, e que $k \leq n$. Então $\langle A_0, \ldots, A_k \rangle$ é uma sequência de formação para A_k.*

Prova. Exercício. □

Teorema 6.29. $\mathrm{Frm}(\mathscr{L})$ *é o conjunto de todas as $\mathscr{L}$-cadeias A tais que existe uma sequência de formação de fórmulas para A.*

Prova. Seja F o conjunto de todas as cadeias de símbolos na linguagem $\mathscr{L}$ que têm uma sequência de formação. Vimos em Proposição 6.27 que $\mathrm{Frm}(\mathscr{L}) \subseteq F$, então agora provamos a recíproca.

Suponha que A tem uma sequência de formação $\langle A_0, \ldots, A_n \rangle$. Provamos que $A \in \mathrm{Frm}(\mathscr{L})$ por indução forte sobre n. Nossa hipótese de indução é que toda cadeia de símbolos com uma sequência de formação de comprimento $m < n$ está em $\mathrm{Frm}(\mathscr{L})$. Pela definição de uma sequência de formação, ou $A \equiv A_n$ é atômica ou devem existir $j, k < n$ tais que um dos seguintes casos vale:

1. $A \equiv \neg A_j$.

2. $A \equiv (A_j \wedge A_k)$.
3. $A \equiv (A_j \vee A_k)$.
4. $A \equiv (A_j \to A_k)$.
5. $A \equiv \forall x\, A_j$.
6. $A \equiv \exists x\, A_j$.

Agora raciocinamos por casos. Se A é atômica então $A_n \in \mathrm{Frm}(\mathscr{L}_0)$. Suponha, em vez disso, que $A \equiv (A_j \wedge A_k)$. Por Lema 6.28, $\langle A_0, \ldots, A_j \rangle$ e $\langle A_0, \ldots, A_k \rangle$ são sequências de formação para A_j e A_k, respectivamente. Como estas são subsequências iniciais próprias da sequência de formação para A, ambas têm comprimento menor que n. Portanto, pela hipótese de indução, A_j e A_k estão em $\mathrm{Frm}(\mathscr{L}_0)$, e pela definição de uma fórmula, $(A_j \wedge A_k)$ também está. Os outros casos seguem por raciocínio análogo. □

Sequências de formação para termos têm propriedades semelhantes às das fórmulas.

Proposição 6.30. $\mathrm{Trm}(\mathscr{L})$ *é o conjunto de todas as* $\mathscr{L}$*-cadeias* t *tais que existe uma sequência de formação de termos para* t.

Prova. Exercício. □

Há dois tipos de "lixo" que podem aparecer em sequências de formação: elementos repetidos, e elementos que são irrelevantes para a construção da fórmula ou termo. Podemos eliminar ambos olhando para sequências de formação mínimas.

Definição 6.31 (Sequências de formação mínimas). Uma sequência de formação $\langle A_0, \ldots, A_n \rangle$ para uma fórmula A é uma *sequência de formação mínima* para A se, para toda outra sequência de formação s para A, o comprimento de s é maior ou igual a $n + 1$.

Similarmente, uma sequência de formação $\langle t_0, \ldots, t_n \rangle$ para um termo t é uma *sequência de formação mínima* para t se, para toda outra sequência de formação s para t, o comprimento de s é maior ou igual a $n+1$.

Note que uma fórmula ou termo pode ter mais de uma sequência de formação mínima, mas elas conterão exatamente as mesmas cadeias.

Proposição 6.32. *As seguintes condições são equivalentes:*

1. *B é uma sub-fórmula de A.*
2. *B ocorre em toda sequência de formação de A.*
3. *B ocorre em uma sequência de formação mínima de A.*

Prova. Exercício. □

6.8 Variáveis Livres e Sentenças

Definição 6.33 (Ocorrências livres de uma variável). As ocorrências *livres* de uma variável em uma fórmula são definidas indutivamente como segue:

1. A é atômica: todas as ocorrências de variável em A são livres.
2. $A \equiv \neg B$: as ocorrências livres de variável de A são exatamente as de B.
3. $A \equiv (B * C)$: as ocorrências livres de variável de A são as em B juntamente com as em C.
4. $A \equiv \forall x\, B$: as ocorrências livres de variável em A são todas as em B exceto as ocorrências de x.

5. $A \equiv \exists x\, B$: as ocorrências livres de variável em A são todas as em B exceto as ocorrências de x.

Definição 6.34 (Variáveis Ligadas). Uma ocorrência de uma variável em uma fórmula A é *ligada* se não é livre.

Definição 6.35 (Escopo). Se $\forall x\, B$ é uma ocorrência de uma subfórmula em uma fórmula A, então a ocorrência correspondente de B em A é chamada de *escopo* da ocorrência correspondente de $\forall x$. Analogamente para $\exists x$.

Se B é o escopo de uma ocorrência de quantificador $\forall x$ ou $\exists x$ em A, então as ocorrências livres de x em B são ligadas em $\forall x\, B$ e $\exists x\, B$. Dizemos que essas ocorrências são *ligadas pela* ocorrência de quantificador mencionada.

Exemplo 6.36. Considere a seguinte fórmula:

$$\exists v_0 \underbrace{A_0^2(v_0, v_1)}_{B}$$

B representa o escopo de $\exists v_0$. O quantificador liga a ocorrência de v_0 em B, mas não liga a ocorrência de v_1. Então v_1 é uma variável livre neste caso.

Agora podemos ver como isso poderia funcionar em uma fórmula A mais complicada:

$$\forall v_0 \underbrace{(A_0^1(v_0) \to A_0^2(v_0, v_1))}_{B} \to \exists v_1 \underbrace{(A_1^2(v_0, v_1) \vee \forall v_0 \overbrace{\neg A_1^1(v_0)}^{D})}_{C}$$

B é o escopo do primeiro $\forall v_0$, C é o escopo de $\exists v_1$, e D é o escopo do segundo $\forall v_0$. O primeiro $\forall v_0$ liga as ocorrências de v_0 em B, $\exists v_1$ liga a ocorrência de v_1 em C, e o segundo $\forall v_0$ liga a

ocorrência de v_0 em D. A primeira ocorrência de v_1 e a quarta ocorrência de v_0 são livres em A. A última ocorrência de v_0 é livre em D, mas ligada em C e A.

Definição 6.37 (Sentença). Uma fórmula A é uma *sentença* sse não contém ocorrências livres de variáveis.

6.9 Substituição

Definição 6.38 (Substituição em um termo). Definimos $s[t/x]$, o resultado de *substituir* t por toda ocorrência de x em s, recursivamente:

1. $s \equiv c$: $s[t/x]$ é simplesmente s.
2. $s \equiv y$: $s[t/x]$ é também simplesmente s, desde que y seja uma variável e $y \not\equiv x$.
3. $s \equiv x$: $s[t/x]$ é t.
4. $s \equiv f(t_1, \ldots, t_n)$: $s[t/x]$ é $f(t_1[t/x], \ldots, t_n[t/x])$.

Definição 6.39. Um termo t está *livre para* x em A se nenhuma das ocorrências livres de x em A ocorre no escopo de um quantificador que liga uma variável em t.

Exemplo 6.40.

1. v_8 está livre para v_1 em $\exists v_3 A_4^2(v_3, v_1)$
2. $f_1^2(v_1, v_2)$ *não* está livre para v_0 em $\forall v_2 A_4^2(v_0, v_2)$

Definição 6.41 (Substituição em uma fórmula). Se A é uma fórmula, x é uma variável, e t é um termo livre para x em A, então $A[t/x]$ é o resultado de substituir t por todas as ocorrências livres de x em A.

1. $A \equiv \bot$: $A[t/x]$ é $\bot$.
2. $A \equiv P(t_1, \ldots, t_n)$: $A[t/x]$ é $P(t_1[t/x], \ldots, t_n[t/x])$.
3. $A \equiv t_1 = t_2$: $A[t/x]$ é $t_1[t/x] = t_2[t/x]$.
4. $A \equiv \neg B$: $A[t/x]$ é $\neg B[t/x]$.
5. $A \equiv (B \wedge C)$: $A[t/x]$ é $(B[t/x] \wedge C[t/x])$.
6. $A \equiv (B \vee C)$: $A[t/x]$ é $(B[t/x] \vee C[t/x])$.
7. $A \equiv (B \to C)$: $A[t/x]$ é $(B[t/x] \to C[t/x])$.
8. $A \equiv \forall y\, B$: $A[t/x]$ é $\forall y\, B[t/x]$, desde que y seja uma variável diferente de x; caso contrário $A[t/x]$ é simplesmente A.
9. $A \equiv \exists y\, B$: $A[t/x]$ é $\exists y\, B[t/x]$, desde que y seja uma variável diferente de x; caso contrário $A[t/x]$ é simplesmente A.

Note que a substituição pode ser vácua: Se x não ocorre de modo algum em A, então $A[t/x]$ é simplesmente A.

A restrição de que t deve estar livre para x em A é necessária para excluir casos como o seguinte. Se $A \equiv \exists y\, x < y$ e $t \equiv y$, então $A[t/x]$ seria $\exists y\, y < y$. Neste caso a variável livre y é "capturada" pelo quantificador $\exists y$ ao substituir, e isso é indesejável. Por exemplo, gostaríamos que fosse o caso que, sempre que $\forall x\, B$ vale, $B[t/x]$ também vale. Mas considere $\forall x \exists y\, x < y$ (aqui B é $\exists y\, x < y$). É uma sentença que é verdadeira a respeito, por exemplo, dos números naturais: para todo número x há um número y maior que ele. Se permitíssemos y como uma possível substituição para x, terminaríamos com $B[y/x] \equiv \exists y\, y < y$, que é falsa.

Prevenimos isso exigindo que nenhuma das variáveis livres em t acabe sendo ligada por um quantificador em A.

Frequentemente usamos a seguinte convenção para evitar notação incômoda: Se A é uma fórmula que pode conter a variável x livre, também escrevemos $A(x)$ para indicar isso. Quando está claro qual A e x temos em mente, e t é um termo (suposto estar livre para x em $A(x)$), então escrevemos $A(t)$ como abreviação de $A[t/x]$. Assim, por exemplo, poderíamos dizer, "chamamos $A(t)$ de uma instância de $\forall x\, A(x)$." Com isso queremos dizer que se A é qualquer fórmula, x uma variável, e t um termo que está livre para x em A, então $A[t/x]$ é uma instância de $\forall x\, A$.

Resumo

Uma **linguagem de primeira ordem** consiste em símbolos de **constante**, **função** e **predicado**. Os símbolos de função e de constante recebem um número especificado de argumentos. Na **linguagem da aritmética**, por exemplo, temos um único símbolo de constante 0, um símbolo de função unário $'$, dois símbolos de função binários $+$ e $\times$, e um símbolo de predicado binário $<$. A partir de **variáveis** e de símbolos de constante e de função, formamos os **termos** de uma linguagem. A partir dos termos de uma linguagem junto com seus símbolos de predicado, bem como o **símbolo de identidade** $=$, formamos as **fórmulas atômicas**. E, por sua vez, a partir delas, usando os conectivos lógicos $\neg$, $\vee$, $\wedge$, $\rightarrow$, $\leftrightarrow$ e os quantificadores $\forall$ e $\exists$, formamos suas fórmulas. Como temos o cuidado de sempre incluir os parênteses necessários no processo de formação de termos e fórmulas, há sempre exatamente um modo de ler uma fórmula. Isso torna possível definir coisas por indução sobre a estrutura das fórmulas.

As ocorrências de variáveis em fórmulas são às vezes governadas por um quantificador correspondente: se uma variável ocorre no **escopo** de um quantificador, ela é considerada **ligada**; caso contrário, **livre**. Esses conceitos têm todos definições indutivas,

e também definimos indutivamente a operação de **substituição** de um termo por uma variável em uma fórmula. As fórmulas sem ocorrências de variáveis livres são chamadas de **sentenças**.

Problemas

Problema 6.1. Prove Lema 6.8.

Problema 6.2. Prove que, para qualquer termo t, $l(t) = r(t)$.

Problema 6.3. Prove Lema 6.12.

Problema 6.4. Prove Proposição 6.13 (Dica: Formule e prove uma versão de Lema 6.12 para termos.)

Problema 6.5. Prove Proposição 6.19.

Problema 6.6. Prove Proposição 6.20.

Problema 6.7. Prove Lema 6.28.

Problema 6.8. Prove Proposição 6.30. Dica: use uma estratégia semelhante à usada na prova de Teorema 6.29.

Problema 6.9. Prove Proposição 6.32.

Problema 6.10. Dê uma definição indutiva das ocorrências de variáveis ligadas nos moldes de Definição 6.33.

CAPÍTULO 7

Semântica da Lógica de Primeira Ordem

7.1 Introdução

Dar o significado das expressões é o domínio da semântica. O conceito central em semântica é o de satisfação em uma estrutura. Uma estrutura dá significado aos blocos de construção da linguagem: um domínio é um conjunto não vazio de objetos. Os quantificadores são interpretados como variando sobre esse domínio, aos símbolos de constante são atribuídos elementos no domínio, aos símbolos de função são atribuídas funções do domínio para si mesmo, e aos predicados são atribuídas relações sobre o domínio. O domínio junto com as atribuições ao vocabulário básico constitui uma estrutura. Variáveis podem aparecer em fórmulas, e para dar uma semântica, também temos que atribuir elementos do domínio a elas—isto é uma atribuição de

variáveis. A relação de satisfação, finalmente, reúne tudo isso. Uma fórmula pode ser satisfeita em uma estrutura M relativamente a uma atribuição de variáveis s, escrita como $M, s \vDash A$. Essa relação também é definida por indução sobre a estrutura de A, usando as tabelas de verdade para os conectivos lógicos para definir, por exemplo, a satisfação de $(A \wedge B)$ em termos da satisfação (ou não) de A e B. Acontece então que a atribuição de variáveis é irrelevante se uma fórmula A é uma sentença, isto é, não tem variáveis livres, e assim podemos falar de sentenças sendo simplesmente satisfeitas (ou não) em estruturas.

Com base na relação de satisfação $M \vDash A$ para sentenças podemos então definir as noções semânticas básicas de validade, consequência e satisfatibilidade. Uma sentença é válida, $\vDash A$, se toda estrutura a satisfaz. Ela é consequência de um conjunto de sentenças, $\Gamma \vDash A$, se toda estrutura que satisfaz todas as sentenças em Γ também satisfaz A. E um conjunto de sentenças é satisfatível se alguma estrutura satisfaz todas as sentenças nele ao mesmo tempo. Como fórmulas são definidas indutivamente, e a satisfação é por sua vez definida por indução sobre a estrutura das fórmulas, podemos usar indução para provar propriedades de nossa semântica e para relacionar as noções semânticas definidas.

7.2 Estruturas para Linguagens de Primeira Ordem

As linguagens de primeira ordem são, por si mesmas, *não interpretadas:* os símbolos de constante, símbolos de função e predicados não têm nenhum significado específico associado a eles. Os significados são dados especificando-se uma *estrutura.* Ela especifica o *domínio*, isto é, os objetos que os símbolos de constante denotam, sobre os quais os símbolos de função operam e sobre os quais os quantificadores variam. Além disso, ela especifica quais símbolos de constante denotam quais objetos, como um símbolo de função mapeia objetos para objetos, e a quais objetos os predicados

se aplicam. Estruturas são a base para as noções *semânticas* em lógica, por exemplo, a noção de consequência, validade, satisfatibilidade. Elas são chamadas de várias formas na literatura: "estruturas", "interpretações" ou "modelos".

Definição 7.1 (Estruturas). Uma *estrutura* $\boldsymbol{M}$, para uma linguagem $\mathcal{L}$ de lógica de primeira ordem consiste nos seguintes elementos:

1. *Domínio:* um conjunto não vazio, $|\boldsymbol{M}|$
2. *Interpretação dos símbolos de constante:* para cada símbolo de constante c de $\mathcal{L}$, um elemento $c^{\boldsymbol{M}} \in |\boldsymbol{M}|$
3. *Interpretação dos predicados:* para cada predicado R de n lugares de $\mathcal{L}$ (distinto de $=$), uma relação de n lugares $R^{\boldsymbol{M}} \subseteq |\boldsymbol{M}|^n$
4. *Interpretação dos símbolos de função:* para cada símbolo de função f de n lugares de $\mathcal{L}$, uma função de n lugares $f^{\boldsymbol{M}} \colon |\boldsymbol{M}|^n \to |\boldsymbol{M}|$

Exemplo 7.2. Uma estrutura $\boldsymbol{M}$ para a linguagem da aritmética consiste em um conjunto, um elemento de $|\boldsymbol{M}|$, $\mathsf{o}^{\boldsymbol{M}}$, como interpretação do símbolo de constante o, uma função de um lugar $\mathsf{'}^{\boldsymbol{M}} \colon |\boldsymbol{M}| \to |\boldsymbol{M}|$, duas funções de dois lugares $+^{\boldsymbol{M}}$ e $\times^{\boldsymbol{M}}$, ambas $|\boldsymbol{M}|^2 \to |\boldsymbol{M}|$, e uma relação de dois lugares $<^{\boldsymbol{M}} \subseteq |\boldsymbol{M}|^2$.

Um exemplo óbvio de tal estrutura é o seguinte:

1. $|\boldsymbol{N}| = \mathbb{N}$
2. $\mathsf{o}^{\boldsymbol{N}} = 0$
3. $\mathsf{'}^{\boldsymbol{N}}(n) = n + 1$ para todos $n \in \mathbb{N}$
4. $+^{\boldsymbol{N}}(n, m) = n + m$ para todos $n, m \in \mathbb{N}$
5. $\times^{\boldsymbol{N}}(n, m) = n \cdot m$ para todos $n, m \in \mathbb{N}$

6. $<^N = \{\langle n, m\rangle : n \in \mathbb{N}, m \in \mathbb{N}, n < m\}$

A estrutura $\boldsymbol{N}$ para $\mathscr{L}_A$ assim definida é chamada de *modelo padrão da aritmética*, porque interpreta as constantes não lógicas de $\mathscr{L}_A$ exatamente como se esperaria.

No entanto, há muitas outras estruturas possíveis para $\mathscr{L}_A$. Por exemplo, poderíamos tomar como domínio o conjunto $\mathbb{Z}$ dos inteiros em vez de $\mathbb{N}$, e definir as interpretações de 0, $'$, $+$, $\times$, $<$ adequadamente. Mas também podemos definir estruturas para $\mathscr{L}_A$ que não têm nada, nem mesmo remotamente, a ver com números.

Exemplo 7.3. Uma estrutura $\boldsymbol{M}$ para a linguagem $\mathscr{L}_Z$ da teoria dos conjuntos requer apenas um conjunto e uma única relação de dois lugares. Então, tecnicamente, por exemplo, o conjunto das pessoas mais a relação "x é mais velho que y" poderia ser usado como uma estrutura para $\mathscr{L}_Z$, assim como $\mathbb{N}$ junto com $n \geq m$ para $n, m \in \mathbb{N}$.

Uma estrutura particularmente interessante para $\mathscr{L}_Z$ na qual os elementos do domínio são de fato conjuntos, e a interpretação de $\in$ é de fato a relação "x é um elemento de y" é a estrutura $\boldsymbol{HF}$ dos *conjuntos hereditariamente finitos*:

1. $|\boldsymbol{HF}| = \emptyset \cup \wp(\emptyset) \cup \wp(\wp(\emptyset)) \cup \wp(\wp(\wp(\emptyset))) \cup \ldots$;
2. $\in^{\boldsymbol{HF}} = \{\langle x, y\rangle : x, y \in |\boldsymbol{HF}|, x \in y\}$.

As estipulações que fazemos sobre o que conta como uma estrutura impactam nossa lógica. Por exemplo, a escolha de impedir domínios vazios garante, dada a explicação usual de satisfação (ou verdade) para sentenças quantificadas, que $\exists x\,(A(x) \vee \neg A(x))$ é válida—isto é, uma verdade lógica. E a estipulação de que todos os símbolos de constante devem referir-se a um objeto no domínio garante que a generalização existencial é um padrão de inferência correto: $A(a)$, portanto $\exists x\, A(x)$. Se permitíssemos que nomes se referissem a algo fora do domínio, ou que não se referissem, então estaríamos a caminho de uma *lógica livre*, na

qual a generalização existencial requer uma premissa adicional: $A(a)$ e $\exists x\, x = a$, portanto $\exists x\, A(x)$.

7.3 Estruturas Cobertas para Linguagens de Primeira Ordem

Lembre-se de que um termo é *fechado* se não contém variáveis.

Definição 7.4 (Valor de termos fechados). Se t é um termo fechado da linguagem $\mathscr{L}$ e $\boldsymbol{M}$ é uma estrutura para $\mathscr{L}$, o *valor* $\mathrm{Val}^{\boldsymbol{M}}(t)$ é definido como segue:

1. Se t é apenas o símbolo de constante c, então $\mathrm{Val}^{\boldsymbol{M}}(c) = c^{\boldsymbol{M}}$.
2. Se t é da forma $f(t_1, \ldots, t_n)$, então

$$\mathrm{Val}^{\boldsymbol{M}}(t) = f^{\boldsymbol{M}}(\mathrm{Val}^{\boldsymbol{M}}(t_1), \ldots, \mathrm{Val}^{\boldsymbol{M}}(t_n)).$$

Definição 7.5 (Estrutura Coberta). Uma estrutura é *coberta* se todo elemento do domínio é o valor de algum termo fechado.

Exemplo 7.6. Seja $\mathscr{L}$ a linguagem com os símbolos de constante zero, um, dois, ..., o predicado binário $<$, e os símbolos de função binários $+$ e $\times$. Então uma estrutura $\boldsymbol{M}$ para $\mathscr{L}$ é aquela com domínio $|\boldsymbol{M}| = \{0, 1, 2, \ldots\}$ e atribuições $\mathit{zero}^{\boldsymbol{M}} = 0$, $\mathit{um}^{\boldsymbol{M}} = 1$, $\mathit{dois}^{\boldsymbol{M}} = 2$, e assim por diante. Para o símbolo de relação binário $<$, o conjunto $<^{\boldsymbol{M}}$ é o conjunto de todos os pares $\langle c_1, c_2\rangle \in |\boldsymbol{M}|^2$ tais que c_1 é menor que c_2: por exemplo, $\langle 1, 3\rangle \in <^{\boldsymbol{M}}$ mas $\langle 2, 2\rangle \notin <^{\boldsymbol{M}}$. Para o símbolo de função binário $+$, defina $+^{\boldsymbol{M}}$ da maneira usual—por exemplo, $+^{\boldsymbol{M}}(2, 3)$ resulta em 5, e analogamente para o símbolo de função binário $\times$. Portanto, o valor de quatro é apenas 4, e o valor de $\times(\mathit{dois}, +(\mathit{tres}, \mathit{zero}))$ (ou

em notação infixa, $dois \times (tres + zero)$) é

$$\begin{aligned}
\mathrm{Val}^{\mathcal{M}}(\times(dois, +(tres, zero))) &= \\
&= \times^{\mathcal{M}}(\mathrm{Val}^{\mathcal{M}}(dois), \mathrm{Val}^{\mathcal{M}}(+(tres, zero))) \\
&= \times^{\mathcal{M}}(\mathrm{Val}^{\mathcal{M}}(dois), +^{\mathcal{M}}(\mathrm{Val}^{\mathcal{M}}(tres), \mathrm{Val}^{\mathcal{M}}(zero))) \\
&= \times^{\mathcal{M}}(dois^{\mathcal{M}}, +^{\mathcal{M}}(tres^{\mathcal{M}}, zero^{\mathcal{M}})) \\
&= \times^{\mathcal{M}}(2, +^{\mathcal{M}}(3, 0)) \\
&= \times^{\mathcal{M}}(2, 3) \\
&= 6
\end{aligned}$$

7.4 Satisfação de uma Fórmula em uma Estrutura

A noção básica que relaciona expressões como termos e fórmulas, por um lado, e estruturas, por outro, são as de *valor* de um termo e *satisfação* de uma fórmula. Informalmente, o valor de um termo é um elemento de uma estrutura—se o termo é apenas uma constante, seu valor é o objeto atribuído à constante pela estrutura, e se ele é construído usando símbolos de função, o valor é calculado a partir dos valores das constantes e das funções atribuídas às funções no termo. Uma fórmula é *satisfeita* em uma estrutura se a interpretação dada aos predicados torna a fórmula verdadeira no domínio da estrutura. Essa noção de satisfação é especificada indutivamente: a especificação da estrutura afirma diretamente quando fórmulas atômicas são satisfeitas, e definimos quando uma fórmula complexa é satisfeita dependendo do conectivo principal ou quantificador e de se as subfórmulas imediatas são ou não satisfeitas.

O caso dos quantificadores aqui é um pouco delicado, pois a subfórmula imediata de uma fórmula quantificada tem uma variável livre, e estruturas não especificam os valores das variáveis. Para lidar com essa dificuldade, também introduzimos *atribuições de variáveis* e definimos a satisfação não em relação a uma estru-

tura sozinha, mas em relação a uma estrutura mais uma atribuição de variáveis.

Definição 7.7 (Atribuição de Variáveis). Uma *atribuição de variáveis* s para uma estrutura $\boldsymbol{M}$ é uma função que mapeia cada variável para um elemento de $|\boldsymbol{M}|$, isto é, $s\colon \mathrm{Var} \to |\boldsymbol{M}|$.

Uma estrutura atribui um valor a cada símbolo de constante, e uma atribuição de variáveis a cada variável. Mas queremos usar termos construídos a partir deles para também nomear elementos do domínio. Para isso, definimos o valor dos termos indutivamente. Para símbolos de constante e variáveis, o valor é exatamente como a estrutura ou a atribuição de variáveis o especifica; para termos mais complexos, ele é calculado recursivamente usando as funções que a estrutura atribui aos símbolos de função.

Definição 7.8 (Valor de Termos). Se t é um termo da linguagem $\mathscr{L}$, $\boldsymbol{M}$ é uma estrutura para $\mathscr{L}$, e s é uma atribuição de variáveis para $\boldsymbol{M}$, o *valor* $\mathrm{Val}^{\boldsymbol{M}}_s(t)$ é definido como segue:

1. $t \equiv c$: $\mathrm{Val}^{\boldsymbol{M}}_s(t) = c^{\boldsymbol{M}}$.
2. $t \equiv x$: $\mathrm{Val}^{\boldsymbol{M}}_s(t) = s(x)$.
3. $t \equiv f(t_1, \ldots, t_n)$:

$$\mathrm{Val}^{\boldsymbol{M}}_s(t) = f^{\boldsymbol{M}}(\mathrm{Val}^{\boldsymbol{M}}_s(t_1), \ldots, \mathrm{Val}^{\boldsymbol{M}}_s(t_n)).$$

Definição 7.9 (x-Variante). Se s é uma atribuição de variáveis para uma estrutura $\boldsymbol{M}$, então qualquer atribuição de variáveis s' para $\boldsymbol{M}$ que difere de s no máximo no que atribui a x é chamada de *x-variante* de s. Se s' é uma x-variante de s escrevemos $s' \sim_x s$.

Note que uma x-variante de uma atribuição s não *precisa* atribuir algo diferente a x. De fato, toda atribuição conta como uma x-variante de si mesma.

Definição 7.10. Se s é uma atribuição de variáveis para uma estrutura $\boldsymbol{M}$ e $m \in |\boldsymbol{M}|$, então a atribuição $s[m/x]$ é a atribuição de variáveis definida por

$$s[m/x](y) = \begin{cases} m & \text{se } y \equiv x \\ s(y) & \text{caso contrário.} \end{cases}$$

Em outras palavras, $s[m/x]$ é a x-variante particular de s que atribui o elemento m do domínio a x, e atribui as mesmas coisas às variáveis diferentes de x que s atribui.

Definição 7.11 (Satisfação). A satisfação de uma fórmula A em uma estrutura $\boldsymbol{M}$ relativamente a uma atribuição de variáveis s, em símbolos: $\boldsymbol{M}, s \vDash A$, é definida recursivamente como segue. (Escrevemos $\boldsymbol{M}, s \nvDash A$ para significar "não $\boldsymbol{M}, s \vDash A$".)

1. $A \equiv \bot$: $\boldsymbol{M}, s \nvDash A$.
2. $A \equiv R(t_1, \ldots, t_n)$: $\boldsymbol{M}, s \vDash A$ sse $\langle \mathrm{Val}^{\boldsymbol{M}}_s(t_1), \ldots, \mathrm{Val}^{\boldsymbol{M}}_s(t_n)\rangle \in R^{\boldsymbol{M}}$.
3. $A \equiv t_1 = t_2$: $\boldsymbol{M}, s \vDash A$ sse $\mathrm{Val}^{\boldsymbol{M}}_s(t_1) = \mathrm{Val}^{\boldsymbol{M}}_s(t_2)$.
4. $A \equiv \neg B$: $\boldsymbol{M}, s \vDash A$ sse $\boldsymbol{M}, s \nvDash B$.
5. $A \equiv (B \wedge C)$: $\boldsymbol{M}, s \vDash A$ sse $\boldsymbol{M}, s \vDash B$ e $\boldsymbol{M}, s \vDash C$.
6. $A \equiv (B \vee C)$: $\boldsymbol{M}, s \vDash A$ sse $\boldsymbol{M}, s \vDash B$ ou $\boldsymbol{M}, s \vDash C$ (ou ambos).
7. $A \equiv (B \to C)$: $\boldsymbol{M}, s \vDash A$ sse $\boldsymbol{M}, s \nvDash B$ ou $\boldsymbol{M}, s \vDash C$ (ou ambos).
8. $A \equiv \forall x\, B$: $\boldsymbol{M}, s \vDash A$ sse para todo elemento $m \in |\boldsymbol{M}|$, $\boldsymbol{M}, s[m/x] \vDash B$.
9. $A \equiv \exists x\, B$: $\boldsymbol{M}, s \vDash A$ sse para pelo menos um elemento $m \in |\boldsymbol{M}|$, $\boldsymbol{M}, s[m/x] \vDash B$.

As atribuições de variáveis são importantes nas últimas duas cláusulas. Não podemos definir a satisfação de $\forall x\, B(x)$ por "para todo $m \in |M|$, $M \vDash B(m)$". Não podemos definir a satisfação de $\exists x\, B(x)$ por "para pelo menos um $m \in |M|$, $M \vDash B(m)$". A razão é que, se $m \in |M|$, ele não é um símbolo da linguagem, e portanto $B(m)$ não é uma fórmula (isto é, $B[m/x]$ é indefinido). Também não podemos supor que temos símbolos de constante ou termos disponíveis que nomeiem cada elemento de M, já que não há nada na definição de estruturas que o exija. Na linguagem padrão, o conjunto de símbolos de constante é infinito contável, então se $|M|$ não é contável não há sequer símbolos de constante suficientes para nomear cada objeto.

Resolvemos esse problema introduzindo atribuições de variáveis, que nos permitem ligar variáveis diretamente a elementos do domínio. Então, em vez de dizer que, por exemplo, $\exists x\, B(x)$ é satisfeita em M sse para pelo menos um $m \in |M|$, dizemos que ela é satisfeita em M *relativamente a* s sse $B(x)$ é satisfeita relativamente a $s[m/x]$ para pelo menos um $m \in |M|$.

Exemplo 7.12. Seja $\mathscr{L} = \{a, b, f, R\}$ onde a e b são símbolos de constante, f é um símbolo de função de dois lugares, e R é um predicado de dois lugares. Considere a estrutura M definida por:

1. $|M| = \{1, 2, 3, 4\}$
2. $a^M = 1$
3. $b^M = 2$
4. $f^M(x, y) = x + y$ se $x + y \leq 3$ e $= 3$ caso contrário.
5. $R^M = \{\langle 1,1\rangle, \langle 1,2\rangle, \langle 2,3\rangle, \langle 2,4\rangle\}$

A função $s(x) = 1$ que atribui $1 \in |M|$ a cada variável é uma atribuição de variáveis para M.

Então

$$\mathrm{Val}_s^M(f(a,b)) = f^M(\mathrm{Val}_s^M(a), \mathrm{Val}_s^M(b)).$$

Como a e b são símbolos de constante, $\mathrm{Val}_s^M(a) = a^M = 1$ e $\mathrm{Val}_s^M(b) = b^M = 2$. Então

$$\mathrm{Val}_s^M(f(a,b)) = f^M(1,2) = 1 + 2 = 3.$$

Para calcular o valor de $f(f(a,b),a)$ temos que considerar

$$\mathrm{Val}_s^M(f(f(a,b),a)) = f^M(\mathrm{Val}_s^M(f(a,b)),\mathrm{Val}_s^M(a)) = f^M(3,1) = 3,$$

já que $3+1 > 3$. Como $s(x) = 1$ e $\mathrm{Val}_s^M(x) = s(x)$, também temos

$$\mathrm{Val}_s^M(f(f(a,b),x)) = f^M(\mathrm{Val}_s^M(f(a,b)),\mathrm{Val}_s^M(x)) = f^M(3,1) = 3,$$

Uma fórmula atômica $R(t_1,t_2)$ é satisfeita se a tupla de valores de seus argumentos, isto é, $\langle \mathrm{Val}_s^M(t_1), \mathrm{Val}_s^M(t_2)\rangle$, é um elemento de R^M. Assim, por exemplo, temos $\boldsymbol{M}, s \vDash R(b,f(a,b))$ já que $\langle \mathrm{Val}^M(b), \mathrm{Val}^M(f(a,b))\rangle = \langle 2,3\rangle \in R^M$, mas $\boldsymbol{M}, s \nvDash R(x,f(a,b))$ já que $\langle 1,3\rangle \notin R^M[s]$.

Para determinar se uma fórmula não atômica A é satisfeita, aplicam-se as cláusulas da definição indutiva que se aplicam ao conectivo principal. Por exemplo, o conectivo principal em $R(a,a) \to (R(b,x) \vee R(x,b))$ é o $\to$, e

$$\begin{aligned}&\boldsymbol{M}, s \vDash R(a,a) \to (R(b,x) \vee R(x,b)) \text{ sse}\\ &\quad \boldsymbol{M}, s \nvDash R(a,a) \text{ ou } \boldsymbol{M}, s \vDash R(b,x) \vee R(x,b)\end{aligned}$$

Como $\boldsymbol{M}, s \vDash R(a,a)$ (porque $\langle 1,1\rangle \in R^M$) ainda não podemos determinar a resposta e devemos primeiro descobrir se $\boldsymbol{M}, s \vDash R(b,x) \vee R(x,b)$:

$$\begin{aligned}&\boldsymbol{M}, s \vDash R(b,x) \vee R(x,b) \text{ sse}\\ &\quad \boldsymbol{M}, s \vDash R(b,x) \text{ ou } \boldsymbol{M}, s \vDash R(x,b)\end{aligned}$$

E este é o caso, já que $\boldsymbol{M}, s \vDash R(x,b)$ (porque $\langle 1,2\rangle \in R^M$).

Lembre-se de que uma x-variante de s é uma atribuição de variáveis que difere de s no máximo no que atribui a x. Para cada elemento de $|M|$, há uma x-variante de s:

$$s_1 = s[1/x], \qquad s_2 = s[2/x],$$
$$s_3 = s[3/x], \qquad s_4 = s[4/x].$$

Assim, por exemplo, $s_2(x) = 2$ e $s_2(y) = s(y) = 1$ para todas as variáveis y diferentes de x. Estas são todas as x-variantes de s para a estrutura M, já que $|M| = \{1,2,3,4\}$. Note, em particular, que $s_1 = s$ (s é sempre uma x-variante de si mesma).

Para determinar se uma fórmula quantificada existencialmente $\exists x\, A(x)$ é satisfeita, temos que determinar se $M, s[m/x] \vDash A(x)$ para pelo menos um $m \in |M|$. Assim,

$$M, s \vDash \exists x\, (R(b,x) \vee R(x,b)),$$

já que $M, s[1/x] \vDash R(b,x) \vee R(x,b)$ ($s[3/x]$ também serviria). Mas,

$$M, s \nvDash \exists x\, (R(b,x) \wedge R(x,b))$$

já que, qualquer que seja o $m \in |M|$ que escolhamos, $M, s[m/x] \nvDash R(b,x) \wedge R(x,b)$.

Para determinar se uma fórmula quantificada universalmente $\forall x\, A(x)$ é satisfeita, temos que determinar se $M, s[m/x] \vDash A(x)$ para todo $m \in |M|$. Assim,

$$M, s \vDash \forall x\, (R(x,a) \rightarrow R(a,x)),$$

já que $M, s[m/x] \vDash R(x,a) \rightarrow R(a,x)$ para todo $m \in |M|$. Para $m = 1$, temos $M, s[1/x] \vDash R(a,x)$ então o consequente é verdadeiro; para $m = 2$, 3, e 4, temos $M, s[m/x] \nvDash R(x,a)$, então o antecedente é falso. Mas,

$$M, s \nvDash \forall x\, (R(a,x) \rightarrow R(x,a))$$

já que $M, s[2/x] \nvDash R(a,x) \rightarrow R(x,a)$ (porque $M, s[2/x] \vDash R(a,x)$ e $M, s[2/x] \nvDash R(x,a)$).

Para um caso mais complicado, considere

$$\forall x\,(R(a,x) \rightarrow \exists y\,R(x,y)).$$

Como $M, s[3/x] \nvDash R(a,x)$ e $M, s[4/x] \nvDash R(a,x)$, os casos interessantes em que temos que nos preocupar com o consequente do condicional são apenas $m = 1$ e $= 2$. $M, s[1/x] \vDash \exists y\,R(x,y)$ vale? Vale se há pelo menos um $n \in |M|$ tal que $M, s[1/x][n/y] \vDash R(x,y)$. De fato, se tomarmos $n = 1$, temos $s[1/x][n/y] = s[1/y] = s$. Como $s(x) = 1$, $s(y) = 1$, e $\langle 1,1\rangle \in R^M$, a resposta é sim.

Para determinar se $M, s[2/x] \vDash \exists y\,R(x,y)$, temos que olhar para as atribuições de variáveis $s[2/x][n/y]$. Aqui, para $n = 1$, essa atribuição é $s_2 = s[2/x]$, que não satisfaz $R(x,y)$ ($s_2(x) = 2$, $s_2(y) = 1$, e $\langle 2,1\rangle \notin R^M$). No entanto, considere $s[2/x][3/y] = s_2[3/y]$. $M, s_2[3/y] \vDash R(x,y)$ já que $\langle 2,3\rangle \in R^M$, e assim $M, s_2 \vDash \exists y\,R(x,y)$.

Assim, para todo $n \in |M|$, ou $M, s[m/x] \nvDash R(a,x)$ (se $m = 3$, 4) ou $M, s[m/x] \vDash \exists y\,R(x,y)$ (se $m = 1$, 2), e assim

$$M, s \vDash \forall x\,(R(a,x) \rightarrow \exists y\,R(x,y)).$$

Por outro lado,

$$M, s \nvDash \exists x\,(R(a,x) \wedge \forall y\,R(x,y)).$$

Temos $M, s[m/x] \vDash R(a,x)$ apenas para $m = 1$ e $m = 2$. Mas para ambos esses valores de m, há por sua vez um $n \in |M|$, a saber $n = 4$, tal que $M, s[m/x][n/y] \nvDash R(x,y)$ e assim $M, s[m/x] \nvDash \forall y\,R(x,y)$ para $m = 1$ e $m = 2$. Em suma, não há $m \in |M|$ tal que $M, s[m/x] \vDash R(a,x) \wedge \forall y\,R(x,y)$.

7.5 Atribuições de Variáveis

Uma atribuição de variáveis s fornece um valor para *toda* variável—e há infinitas delas. Isso, é claro, não é necessário.

Exigimos que as atribuições de variáveis atribuam valores a todas as variáveis simplesmente porque isso torna as coisas muito mais fáceis. O valor de um termo t, e se uma fórmula A é satisfeita em uma estrutura com respeito a s, dependem apenas das atribuições que s faz às variáveis em t e às variáveis livres de A. Esse é o conteúdo das próximas duas proposições. Para tornar precisa a ideia de "depende de", mostramos que quaisquer duas atribuições de variáveis que concordam em todas as variáveis em t dão o mesmo valor, e que A é satisfeita relativamente a uma sse é satisfeita relativamente à outra se duas atribuições de variáveis concordam em todas as variáveis livres de A.

Proposição 7.13. *Se as variáveis em um termo t estão entre $x_1, \ldots, x_n$, e $s_1(x_i) = s_2(x_i)$ para $i = 1, \ldots, n$, então* $\mathrm{Val}^M_{s_1}(t) = \mathrm{Val}^M_{s_2}(t)$.

Prova. Por indução sobre a complexidade de t. Para o caso base, t pode ser um símbolo de constante ou uma das variáveis $x_1, \ldots, x_n$. Se $t = c$, então $\mathrm{Val}^M_{s_1}(t) = c^M = \mathrm{Val}^M_{s_2}(t)$. Se $t = x_i$, $s_1(x_i) = s_2(x_i)$ pela hipótese da proposição, e assim $\mathrm{Val}^M_{s_1}(t) = s_1(x_i) = s_2(x_i) = \mathrm{Val}^M_{s_2}(t)$.

Para o passo indutivo, suponha que $t = f(t_1, \ldots, t_k)$ e que a afirmação vale para $t_1, \ldots, t_k$. Então

$$\begin{aligned}\mathrm{Val}^M_{s_1}(t) &= \mathrm{Val}^M_{s_1}(f(t_1, \ldots, t_k)) \\ &= f^M(\mathrm{Val}^M_{s_1}(t_1), \ldots, \mathrm{Val}^M_{s_1}(t_k)).\end{aligned}$$

Para $j = 1, \ldots, k$, as variáveis de t_j estão entre $x_1, \ldots, x_n$. Pela hipótese de indução, $\mathrm{Val}^M_{s_1}(t_j) = \mathrm{Val}^M_{s_2}(t_j)$. Assim,

$$\begin{aligned}\mathrm{Val}^M_{s_1}(t) &= \mathrm{Val}^M_{s_1}(f(t_1, \ldots, t_k)) \\ &= f^M(\mathrm{Val}^M_{s_1}(t_1), \ldots, \mathrm{Val}^M_{s_1}(t_k)) \\ &= f^M(\mathrm{Val}^M_{s_2}(t_1), \ldots, \mathrm{Val}^M_{s_2}(t_k)) \\ &= \mathrm{Val}^M_{s_2}(f(t_1, \ldots, t_k)) = \mathrm{Val}^M_{s_2}(t).\end{aligned}$$

□

Proposição 7.14. *Se as variáveis livres em A estão entre $x_1, \ldots, x_n$, e $s_1(x_i) = s_2(x_i)$ para $i = 1, \ldots, n$, então $\boldsymbol{M}, s_1 \vDash A$ sse $\boldsymbol{M}, s_2 \vDash A$.*

Prova. Usamos indução sobre a complexidade de A. Para o caso base, onde A é atômica, A pode ser: $\bot$, $R(t_1, \ldots, t_k)$ para um predicado R de k lugares e termos t_1, $\ldots$, t_k, ou $t_1 = t_2$ para termos t_1 e t_2. Nos dois últimos casos, demonstramos apenas a direção direta do bicondicional, já que a prova da recíproca é simétrica.

1. $A \equiv \bot$: tanto $\boldsymbol{M}, s_1 \nvDash A$ quanto $\boldsymbol{M}, s_2 \nvDash A$.

2. $A \equiv R(t_1, \ldots, t_k)$: seja $\boldsymbol{M}, s_1 \vDash A$. Então

$$\langle \mathrm{Val}^{\boldsymbol{M}}_{s_1}(t_1), \ldots, \mathrm{Val}^{\boldsymbol{M}}_{s_1}(t_k) \rangle \in R^{\boldsymbol{M}}.$$

   Para $i = 1, \ldots, k$, $\mathrm{Val}^{\boldsymbol{M}}_{s_1}(t_i) = \mathrm{Val}^{\boldsymbol{M}}_{s_2}(t_i)$ por Proposição 7.13. Então também temos $\langle \mathrm{Val}^{\boldsymbol{M}}_{s_2}(t_i), \ldots, \mathrm{Val}^{\boldsymbol{M}}_{s_2}(t_k) \rangle \in R^{\boldsymbol{M}}$, e portanto $\boldsymbol{M}, s_2 \vDash A$.

3. $A \equiv t_1 = t_2$: suponha $\boldsymbol{M}, s_1 \vDash A$. Então $\mathrm{Val}^{\boldsymbol{M}}_{s_1}(t_1) = \mathrm{Val}^{\boldsymbol{M}}_{s_1}(t_2)$. Assim,

$$\begin{aligned} \mathrm{Val}^{\boldsymbol{M}}_{s_2}(t_1) &= \mathrm{Val}^{\boldsymbol{M}}_{s_1}(t_1) && \text{(por Proposição 7.13)} \\ &= \mathrm{Val}^{\boldsymbol{M}}_{s_1}(t_2) && \text{(já que } \boldsymbol{M}, s_1 \vDash t_1 = t_2\text{)} \\ &= \mathrm{Val}^{\boldsymbol{M}}_{s_2}(t_2) && \text{(por Proposição 7.13),} \end{aligned}$$

   então $\boldsymbol{M}, s_2 \vDash t_1 = t_2$.

Agora suponha $\boldsymbol{M}, s_1 \vDash B$ sse $\boldsymbol{M}, s_2 \vDash B$ para todas as fórmulas B menos complexas que A. O passo de indução procede por casos determinados pelo operador principal de A. Em cada caso, demonstramos apenas a direção direta do bicondicional; a prova da direção reversa é simétrica. Em todos os casos exceto os dos quantificadores, aplicamos a hipótese de indução às subfórmulas B de A. As variáveis livres de B estão entre as de A. Assim, se s_1 e s_2 concordam nas variáveis livres de A, elas também concordam nas de B, e a hipótese de indução se aplica a B.

1. $A \equiv \neg B$: se $\boldsymbol{M}, s_1 \vDash A$, então $\boldsymbol{M}, s_1 \nvDash B$, então pela hipótese de indução, $\boldsymbol{M}, s_2 \nvDash B$, portanto $\boldsymbol{M}, s_2 \vDash A$.

2. $A \equiv B \wedge C$: Exercício.

3. $A \equiv B \vee C$: se $\boldsymbol{M}, s_1 \vDash A$, então $\boldsymbol{M}, s_1 \vDash B$ ou $\boldsymbol{M}, s_1 \vDash C$. Pela hipótese de indução, $\boldsymbol{M}, s_2 \vDash B$ ou $\boldsymbol{M}, s_2 \vDash C$, então $\boldsymbol{M}, s_2 \vDash A$.

4. $A \equiv B \rightarrow C$: Exercício.

5. $A \equiv \exists x\, B$: se $\boldsymbol{M}, s_1 \vDash A$, há um $m \in |\boldsymbol{M}|$ tal que $\boldsymbol{M}, s_1[m/x] \vDash B$. Sejam $s_1' = s_1[m/x]$ e $s_2' = s_2[m/x]$. As variáveis livres de B estão entre $x_1, \ldots, x_n$, e x. $s_1'(x_i) = s_2'(x_i)$, já que s_1' e s_2' são x-variantes de s_1 e s_2, respectivamente, e por hipótese $s_1(x_i) = s_2(x_i)$. $s_1'(x) = s_2'(x) = m$ pela forma como definimos s_1' e s_2'. Então a hipótese de indução se aplica a B e s_1', s_2', então $\boldsymbol{M}, s_2' \vDash B$. Portanto, já que $s_2' = s_2[m/x]$, há um $m \in |\boldsymbol{M}|$ tal que $\boldsymbol{M}, s_2[m/x] \vDash B$, e assim $\boldsymbol{M}, s_2 \vDash A$.

6. $A \equiv \forall x\, B$: Exercício.

Por indução, obtemos que $\boldsymbol{M}, s_1 \vDash A$ sse $\boldsymbol{M}, s_2 \vDash A$ sempre que as variáveis livres em A estão entre $x_1, \ldots, x_n$ e $s_1(x_i) = s_2(x_i)$ para $i = 1, \ldots, n$. □

Sentenças não têm variáveis livres, então quaisquer duas atribuições de variáveis atribuem as mesmas coisas a todas as (zero) variáveis livres de qualquer sentença. A proposição que acabamos de provar significa, então, que se uma sentença é satisfeita ou não em uma estrutura relativamente a uma atribuição de variáveis é completamente independente da atribuição. Registraremos esse fato. Ele justifica a definição de satisfação de uma sentença em uma estrutura (sem mencionar uma atribuição de variáveis) que segue.

Corolário 7.15. *Se A é uma sentença e s uma atribuição de variáveis, então $\mathfrak{M}, s \vDash A$ sse $\mathfrak{M}, s' \vDash A$ para toda atribuição de variáveis s'.*

Prova. Seja s' uma atribuição de variáveis qualquer. Como A é uma sentença, ela não tem variáveis livres, e assim toda atribuição de variáveis s' trivialmente atribui as mesmas coisas a todas as variáveis livres de A que faz s. Então a condição de Proposição 7.14 é satisfeita, e temos $\mathfrak{M}, s \vDash A$ sse $\mathfrak{M}, s' \vDash A$. □

Definição 7.16. Se A é uma sentença, dizemos que uma estrutura $\mathfrak{M}$ *satisfaz* A, $\mathfrak{M} \vDash A$, sse $\mathfrak{M}, s \vDash A$ para todas as atribuições de variáveis s.

Se $\mathfrak{M} \vDash A$, também dizemos simplesmente que A *é verdadeira em* $\mathfrak{M}$. A noção de satisfação se estende naturalmente de sentenças individuais para conjuntos de sentenças.

Definição 7.17. Se Γ é um conjunto de sentenças Γ, dizemos que uma estrutura $\mathfrak{M}$ *satisfaz* Γ, $\mathfrak{M} \vDash \Gamma$, sse $\mathfrak{M} \vDash A$ para todo $A \in \Gamma$.

Proposição 7.18. *Seja $\mathfrak{M}$ uma estrutura, A uma sentença, e s uma atribuição de variáveis. $\mathfrak{M} \vDash A$ sse $\mathfrak{M}, s \vDash A$.*

Prova. Exercício. □

Proposição 7.19. *Suponha que $A(x)$ contém apenas x livre, e $\mathfrak{M}$ é uma estrutura. Então:*

1. *$\mathfrak{M} \vDash \exists x\, A(x)$ sse $\mathfrak{M}, s \vDash A(x)$ para pelo menos uma atribuição de variáveis s.*
2. *$\mathfrak{M} \vDash \forall x\, A(x)$ sse $\mathfrak{M}, s \vDash A(x)$ para todas as atribuições de variáveis s.*

Prova. Exercício. □

7.6 Extensionalidade

A extensionalidade, às vezes chamada de relevância, pode ser expressa informalmente da seguinte forma: os únicos fatores que influenciam a satisfação de fórmula A em uma estrutura M em relação a uma atribuição de variáveis s são o tamanho do domínio e as atribuições feitas por M e s aos elementos da linguagem que de fato aparecem em A.

Uma consequência imediata da extensionalidade é que, quando duas estruturas M e M' concordam em todos os elementos da linguagem que aparecem em uma sentença A e têm o mesmo domínio, M e M' também devem concordar quanto a se A em si é verdadeira ou não.

Proposição 7.20 (Extensionalidade). *Seja A uma fórmula, e M_1 e M_2 estruturas com $|M_1| = |M_2|$, e s uma atribuição de variáveis sobre $|M_1| = |M_2|$. Se $c^{M_1} = c^{M_2}$, $R^{M_1} = R^{M_2}$, e $f^{M_1} = f^{M_2}$ para todo símbolo de constante c, símbolo de relação R, e símbolo de função f que ocorre em A, então $M_1, s \vDash A$ sse $M_2, s \vDash A$.*

Prova. Primeiro prove (por indução sobre t) que, para todo termo, $\mathrm{Val}_s^{M_1}(t) = \mathrm{Val}_s^{M_2}(t)$. Então prove a proposição por indução sobre A, fazendo uso da afirmação que acabamos de provar para a base da indução (onde A é atômica). □

Corolário 7.21 (Extensionalidade para Sentenças). *Seja A uma sentença e M_1, M_2 como em Proposição 7.20. Então $M_1 \vDash A$ sse $M_2 \vDash A$.*

Prova. Segue de Proposição 7.20 por Corolário 7.15. □

Além disso, o valor de um termo, e se uma estrutura satisfaz ou não uma fórmula, dependem apenas dos valores de seus subtermos.

Proposição 7.22. *Seja M uma estrutura, t e t' termos, e s uma atribuição de variáveis. Então* $\mathrm{Val}^M_s(t[t'/x]) = \mathrm{Val}^M_{s[\mathrm{Val}^M_s(t')/x]}(t)$.

Prova. Por indução sobre t.

1. Se t é uma constante, digamos, $t \equiv c$, então $t[t'/x] = c$, e $\mathrm{Val}^M_s(c) = c^M = \mathrm{Val}^M_{s[\mathrm{Val}^M_s(t')/x]}(c)$.

2. Se t é uma variável diferente de x, digamos, $t \equiv y$, então $t[t'/x] = y$, e $\mathrm{Val}^M_s(y) = \mathrm{Val}^M_{s[\mathrm{Val}^M_s(t')/x]}(y)$ já que $s \sim_x s[\mathrm{Val}^M_s(t')/x]$.

3. Se $t \equiv x$, então $t[t'/x] = t'$. Mas $\mathrm{Val}^M_{s[\mathrm{Val}^M_s(t')/x]}(x) = \mathrm{Val}^M_s(t')$ pela definição de $s[\mathrm{Val}^M_s(t')/x]$.

4. Se $t \equiv f(t_1, \ldots, t_n)$ então temos:

$$
\begin{aligned}
&\mathrm{Val}^M_s(t[t'/x]) = \\
&= \mathrm{Val}^M_s(f(t_1[t'/x], \ldots, t_n[t'/x])) \\
&\qquad \text{pela definição de } t[t'/x] \\
&= f^M(\mathrm{Val}^M_s(t_1[t'/x]), \ldots, \mathrm{Val}^M_s(t_n[t'/x])) \\
&\qquad \text{pela definição de } \mathrm{Val}^M_s(f(\ldots)) \\
&= f^M(\mathrm{Val}^M_{s[\mathrm{Val}^M_s(t')/x]}(t_1), \ldots, \mathrm{Val}^M_{s[\mathrm{Val}^M_s(t')/x]}(t_n)) \\
&\qquad \text{pela hipótese de indução} \\
&= \mathrm{Val}^M_{s[\mathrm{Val}^M_s(t')/x]}(t) \text{ pela definição de } \mathrm{Val}^M_{s[\mathrm{Val}^M_s(t')/x]}(f(\ldots))
\end{aligned}
$$

□

Proposição 7.23. *Seja M uma estrutura, A uma fórmula, t' um termo, e s uma atribuição de variáveis. Então $M, s \vDash A[t'/x]$ sse $M, s[\mathrm{Val}^M_s(t')/x] \vDash A$.*

Prova. Exercício. □

O ponto de Proposições 7.22 e 7.23 é o seguinte. Suponha que temos um termo t ou uma fórmula A e algum termo t', e queremos saber o valor de $t[t'/x]$ ou se $A[t'/x]$ é ou não satisfeita em uma estrutura $\boldsymbol{M}$ em relação a uma atribuição de variáveis s. Então podemos ou realizar a substituição primeiro e depois considerar o valor ou a satisfação em relação a $\boldsymbol{M}$ e s, ou podemos primeiro determinar o valor $m = \mathrm{Val}^{\boldsymbol{M}}_s(t')$ de t' em $\boldsymbol{M}$ em relação a s, mudar a atribuição de variáveis para $s[m/x]$ e então considerar o valor de t em $\boldsymbol{M}$ e $s[m/x]$, ou se $\boldsymbol{M}, s[m/x] \vDash A$. Proposições 7.22 e 7.23 garantem que a resposta será a mesma, independentemente de como o façamos.

7.7 Noções Semânticas

Dada a definição de estruturas para linguagens de primeira ordem, podemos definir algumas propriedades semânticas básicas de sentenças e relações entre elas. A mais simples destas é a noção de *validade* de uma sentença. Uma sentença é válida se é satisfeita em toda estrutura. Sentenças válidas são aquelas que são satisfeitas independentemente de como os símbolos não lógicos nela são interpretados. Sentenças válidas são, portanto, também chamadas de *verdades lógicas*—elas são verdadeiras, isto é, satisfeitas, em qualquer estrutura e, portanto, sua verdade depende apenas dos símbolos lógicos que nelas ocorrem e de sua estrutura sintática, mas não dos símbolos não lógicos ou de sua interpretação.

Definição 7.24 (Validade). Uma sentença A é *válida*, $\vDash A$, sse $\boldsymbol{M} \vDash A$ para toda estrutura $\boldsymbol{M}$.

Definição 7.25 (Consequência). Uma sentença A é *consequência* de um conjunto de sentenças Γ, $\Gamma \vDash A$, sse para toda estrutura $\boldsymbol{M}$ com $\boldsymbol{M} \vDash \Gamma$, $\boldsymbol{M} \vDash A$.

Definição 7.26 (Satisfatibilidade). Um conjunto de sentenças Γ é *satisfatível* se $\boldsymbol{M} \vDash \Gamma$ para alguma estrutura $\boldsymbol{M}$. Se Γ não é satisfatível ele é chamado de *insatisfatível.*

Proposição 7.27. *Uma sentença A é válida sse $\Gamma \vDash A$ para todo conjunto de sentenças Γ.*

Prova. Para a direção direta, seja A válida, e seja Γ um conjunto de sentenças. Seja $\boldsymbol{M}$ uma estrutura tal que $\boldsymbol{M} \vDash \Gamma$. Como A é válida, $\boldsymbol{M} \vDash A$, portanto $\Gamma \vDash A$.

Para a contrapositiva da direção reversa, seja A inválida, então há uma estrutura $\boldsymbol{M}$ com $\boldsymbol{M} \nvDash A$. Quando $\Gamma = \{\top\}$, como $\top$ é válida, $\boldsymbol{M} \vDash \Gamma$. Portanto, há uma estrutura $\boldsymbol{M}$ tal que $\boldsymbol{M} \vDash \Gamma$ mas $\boldsymbol{M} \nvDash A$, portanto A não é consequência de Γ. □

Proposição 7.28. *$\Gamma \vDash A$ sse $\Gamma \cup \{\neg A\}$ é insatisfatível.*

Prova. Para a direção direta, suponha $\Gamma \vDash A$ e suponha, ao contrário, que há uma estrutura $\boldsymbol{M}$ tal que $\boldsymbol{M} \vDash \Gamma \cup \{\neg A\}$. Como $\boldsymbol{M} \vDash \Gamma$ e $\Gamma \vDash A$, $\boldsymbol{M} \vDash A$. Além disso, como $\boldsymbol{M} \vDash \Gamma \cup \{\neg A\}$, $\boldsymbol{M} \vDash \neg A$, então temos tanto $\boldsymbol{M} \vDash A$ quanto $\boldsymbol{M} \nvDash A$, uma contradição. Portanto, não pode haver tal estrutura $\boldsymbol{M}$, então $\Gamma \cup \{\neg A\}$ é insatisfatível.

Para a direção reversa, suponha que $\Gamma \cup \{\neg A\}$ é insatisfatível. Então, para toda estrutura $\boldsymbol{M}$, ou $\boldsymbol{M} \nvDash \Gamma$ ou $\boldsymbol{M} \vDash A$. Portanto, para toda estrutura $\boldsymbol{M}$ com $\boldsymbol{M} \vDash \Gamma$, $\boldsymbol{M} \vDash A$, então $\Gamma \vDash A$. □

Proposição 7.29. *Se $\Gamma \subseteq \Gamma'$ e $\Gamma \vDash A$, então $\Gamma' \vDash A$.*

Prova. Suponha que $\Gamma \subseteq \Gamma'$ e $\Gamma \vDash A$. Seja $\boldsymbol{M}$ uma estrutura tal que $\boldsymbol{M} \vDash \Gamma'$; então $\boldsymbol{M} \vDash \Gamma$, e como $\Gamma \vDash A$, obtemos que $\boldsymbol{M} \vDash A$. Portanto, sempre que $\boldsymbol{M} \vDash \Gamma'$, $\boldsymbol{M} \vDash A$, então $\Gamma' \vDash A$. □

Teorema 7.30 (Teorema da Dedução Semântica). *$\Gamma \cup \{A\} \vDash B$ sse $\Gamma \vDash A \to B$.*

Prova. Para a direção direta, seja $\Gamma \cup \{A\} \vDash B$ e seja $\mathfrak{M}$ uma estrutura tal que $\mathfrak{M} \vDash \Gamma$. Se $\mathfrak{M} \vDash A$, então $\mathfrak{M} \vDash \Gamma \cup \{A\}$, então, como B é consequência de $\Gamma \cup \{A\}$, obtemos $\mathfrak{M} \vDash B$. Portanto, $\mathfrak{M} \vDash A \to B$, então $\Gamma \vDash A \to B$.

Para a direção reversa, seja $\Gamma \vDash A \to B$ e $\mathfrak{M}$ uma estrutura tal que $\mathfrak{M} \vDash \Gamma \cup \{A\}$. Então $\mathfrak{M} \vDash \Gamma$, então $\mathfrak{M} \vDash A \to B$, e como $\mathfrak{M} \vDash A$, $\mathfrak{M} \vDash B$. Portanto, sempre que $\mathfrak{M} \vDash \Gamma \cup \{A\}$, $\mathfrak{M} \vDash B$, então $\Gamma \cup \{A\} \vDash B$. □

Proposição 7.31. *Seja $\mathfrak{M}$ uma estrutura, e $A(x)$ uma fórmula com uma variável livre x, e t um termo fechado. Então:*

1. *$A(t) \vDash \exists x\, A(x)$*
2. *$\forall x\, A(x) \vDash A(t)$*

Prova. 1. Suponha $\mathfrak{M} \vDash A(t)$. Seja s uma atribuição de variáveis com $s(x) = \mathrm{Val}^{\mathfrak{M}}(t)$. Então $\mathfrak{M}, s \vDash A(t)$ já que $A(t)$ é uma sentença. Por Proposição 7.23, $\mathfrak{M}, s \vDash A(x)$. Por Proposição 7.19, $\mathfrak{M} \vDash \exists x\, A(x)$.

2. Exercício. □

Resumo

A **semântica** de uma linguagem de primeira ordem é dada por uma **estrutura** para essa linguagem. Ela consiste em um **domínio** e, a cada símbolo de constante, são atribuídos elementos desse domínio. Os símbolos de função são interpretados por funções e os símbolos de relação por relações sobre o domínio. Uma função do conjunto das variáveis para o domínio é uma **atribuição de variáveis**. A relação de **satisfação** relaciona estruturas, atribuições de variáveis e fórmulas; $\mathfrak{M}, s \vDash A$ é definida

por indução sobre a estrutura de A. $M, s \vDash A$ depende apenas da interpretação dos símbolos que de fato ocorrem em A, e, em particular, não depende de s se A não contém variáveis livres. Assim, se A é uma sentença, $M \vDash A$ se $M, s \vDash A$ para algum (ou todo) s.

A relação de satisfação é a base de todas as noções semânticas. Uma sentença é **válida**, $\vDash A$, se é satisfeita em toda estrutura. Uma sentença A é **consequência** de um conjunto de sentenças Γ, $\Gamma \vDash A$, sse $M \vDash A$ para todas as M que satisfazem toda sentença em Γ. Um conjunto Γ é **satisfatível** sse há alguma estrutura que satisfaz toda sentença em Γ; caso contrário, **insatisfatível**. Essas noções estão inter-relacionadas; por exemplo, $\Gamma \vDash A$ sse $\Gamma \cup \{\neg A\}$ é insatisfatível.

Problemas

Problema 7.1. N, o modelo padrão da aritmética, é coberto? Explique.

Problema 7.2. Seja $\mathscr{L} = \{c, f, A\}$ com um símbolo de constante, um símbolo de função de um lugar e um predicado de dois lugares, e seja a estrutura M dada por

1. $|M| = \{1, 2, 3\}$
2. $c^M = 3$
3. $f^M(1) = 2, f^M(2) = 3, f^M(3) = 2$
4. $A^M = \{\langle 1, 2\rangle, \langle 2, 3\rangle, \langle 3, 3\rangle\}$

(a) Seja $s(v) = 1$ para todas as variáveis v. Descubra se

$$M, s \vDash \exists x\, (A(f(z), c) \to \forall y\, (A(y, x) \vee A(f(y), x)))$$

Explique por que sim ou por que não.

(b) Dê uma estrutura e uma atribuição de variáveis diferentes nas quais a fórmula não seja satisfeita.

Problema 7.3. Complete a prova de Proposição 7.14.

Problema 7.4. Prove Proposição 7.18

Problema 7.5. Prove Proposição 7.19.

Problema 7.6. Suponha que $\mathscr{L}$ é uma linguagem sem símbolos de função. Dada uma estrutura M, c um símbolo de constante e $a \in |M|$, defina $M[a/c]$ como a estrutura que é exatamente como M, exceto que $c^{M[a/c]} = a$. Defina $M \models A$ para sentenças A por:

1. $A \equiv \bot$: não $M \models A$.
2. $A \equiv R(d_1, \ldots, d_n)$: $M \models A$ sse $\langle d_1^M, \ldots, d_n^M \rangle \in R^M$.
3. $A \equiv d_1 = d_2$: $M \models A$ sse $d_1^M = d_2^M$.
4. $A \equiv \neg B$: $M \models A$ sse não $M \models B$.
5. $A \equiv (B \wedge C)$: $M \models A$ sse $M \models B$ e $M \models C$.
6. $A \equiv (B \vee C)$: $M \models A$ sse $M \models B$ ou $M \models C$ (ou ambos).
7. $A \equiv (B \to C)$: $M \models A$ sse não $M \models B$ ou $M \models C$ (ou ambos).
8. $A \equiv \forall x\, B$: $M \models A$ sse para todo $a \in |M|$, $M[a/c] \models B[c/x]$, se c não ocorre em B.
9. $A \equiv \exists x\, B$: $M \models A$ sse há um $a \in |M|$ tal que $M[a/c] \models B[c/x]$, se c não ocorre em B.

Sejam $x_1, \ldots, x_n$ todas as variáveis livres em A, $c_1, \ldots, c_n$ símbolos de constante não presentes em A, $a_1, \ldots, a_n \in |M|$, e $s(x_i) = a_i$.

Mostre que $M, s \vDash A$ sse $M[a_1/c_1, \ldots, a_n/c_n] \models A[c_1/x_1] \ldots [c_n/x_n]$.

(Este problema mostra que é possível dar uma semântica para a lógica de primeira ordem que dispensa atribuições de variáveis.)

Problema 7.7. Suponha que f é um símbolo de função não presente em $A(x,y)$. Mostre que há uma estrutura $\boldsymbol{M}$ tal que $\boldsymbol{M} \vDash \forall x\, \exists y\, A(x,y)$ sse há uma $\boldsymbol{M}'$ tal que $\boldsymbol{M}' \vDash \forall x\, A(x, f(x))$.

(Este problema é um caso especial do que é conhecido como Teorema de Skolem; $\forall x\, A(x, f(x))$ é chamado de *forma normal de Skolem* de $\forall x\, \exists y\, A(x,y)$.)

Problema 7.8. Faça a prova de Proposição 7.20 em detalhe.

Problema 7.9. Prove Proposição 7.23

Problema 7.10.

1. Mostre que $\Gamma \vDash \bot$ sse Γ é insatisfatível.
2. Mostre que $\Gamma \cup \{A\} \vDash \bot$ sse $\Gamma \vDash \neg A$.
3. Suponha que c não ocorre em A nem em Γ. Mostre que $\Gamma \vDash \forall x\, A$ sse $\Gamma \vDash A[c/x]$.

Problema 7.11. Complete a prova de Proposição 7.31.

CAPÍTULO 8

Teorias e Seus Modelos

8.1 Introdução

O desenvolvimento do método axiomático é uma conquista significativa na história da ciência, e é de especial importância na história da matemática. Um desenvolvimento axiomático de um campo envolve o esclarecimento de muitas questões: Do que trata o campo? Quais são os conceitos mais fundamentais? Como eles se relacionam? Todos os conceitos do campo podem ser definidos em termos desses conceitos fundamentais? Que leis esses conceitos obedecem, e devem obedecer?

O método axiomático e a lógica foram feitos um para o outro. A lógica formal fornece as ferramentas para formular teorias axiomáticas, para provar teoremas a partir dos axiomas da teoria de uma maneira precisamente especificada, para estudar as propriedades de todos os sistemas que satisfazem os axiomas de uma maneira sistemática.

Definição 8.1. Um conjunto de sentenças Γ é *fechado* sse, sempre que $\Gamma \models A$ então $A \in \Gamma$. O *fecho* de um conjunto de sentenças Γ é $\{A : \Gamma \models A\}$.

Dizemos que Γ é *axiomatizado por* um conjunto de sentenças Δ se Γ é o fecho de Δ.

Podemos pensar em uma teoria axiomática como o conjunto de sentenças que é axiomatizado por seu conjunto de axiomas Δ. Em outras palavras, quando temos uma linguagem de primeira ordem que contém símbolos não lógicos para os primitivos da ciência desenvolvida axiomaticamente que desejamos estudar, juntamente com um conjunto de sentenças que expressam as leis fundamentais da ciência, podemos pensar na teoria como representada por todas as sentenças nessa linguagem que são consequência dos axiomas. Isso varia de exemplos simples com apenas um único primitivo e axiomas simples, como a teoria das ordens parciais, até teorias complexas como a mecânica newtoniana.

Os fatos lógicos importantes que tornam essa abordagem formal do método axiomático tão importante são os seguintes. Suponha que Γ é um sistema de axiomas para uma teoria, isto é, um conjunto de sentenças.

1. Podemos afirmar precisamente quando um sistema de axiomas captura uma classe pretendida de estruturas. Isto é, se estamos interessados em uma certa classe de estruturas, capturaremos com sucesso essa classe por um sistema de axiomas Γ sse as estruturas são exatamente aquelas $\mathfrak{M}$ tais que $\mathfrak{M} \models \Gamma$.

2. Podemos falhar nesse aspecto porque há $\mathfrak{M}$ tais que $\mathfrak{M} \models \Gamma$, mas $\mathfrak{M}$ não é uma das estruturas que pretendemos. Isso pode nos levar a adicionar axiomas que não são verdadeiros em $\mathfrak{M}$.

3. Se somos bem-sucedidos pelo menos no aspecto de que Γ é verdadeiro em todas as estruturas pretendidas, então uma sentença A é verdadeira em todas as estruturas pretendidas sempre que $\Gamma \models A$. Assim, podemos usar ferramentas lógicas (como métodos de derivação) para mostrar que

sentenças são verdadeiras em todas as estruturas pretendidas simplesmente mostrando que elas são consequência dos axiomas.

4. Às vezes não temos estruturas pretendidas em mente, mas em vez disso partimos dos próprios axiomas: começamos com alguns primitivos que queremos que satisfaçam certas leis que codificamos em um sistema de axiomas. Uma coisa que gostaríamos de verificar imediatamente é que os axiomas não se contradizem: se o fizerem, não pode haver conceitos que obedeçam a essas leis, e teremos tentado estabelecer uma teoria incoerente. Podemos verificar que isso não acontece encontrando um modelo de Γ. E se há modelos da nossa teoria, podemos usar métodos lógicos para investigá-los, e também podemos usar métodos lógicos para construir modelos.

5. A independência dos axiomas é, igualmente, uma questão importante. Pode acontecer que um dos axiomas seja, na verdade, uma consequência dos outros, e seja, portanto, redundante. Podemos provar que um axioma A em Γ é redundante provando $\Gamma \setminus \{A\} \vDash A$. Também podemos provar que um axioma não é redundante mostrando que $(\Gamma \setminus \{A\}) \cup \{\neg A\}$ é satisfatível. Por exemplo, foi assim que se mostrou que o postulado das paralelas é independente dos outros axiomas da geometria.

6. Outra questão importante é a da definibilidade de conceitos em uma teoria: A escolha da linguagem determina em que consistem os modelos de uma teoria. Mas nem todo aspecto de uma teoria deve ser representado separadamente em seus modelos. Por exemplo, toda ordenação $\leq$ determina uma ordenação estrita correspondente $<$—dada uma, nós podemos definir a outra. Então não é necessário que um modelo de uma teoria que envolve tal ordem deva conter *também* a ordenação estrita correspondente. Quando é o caso, em geral, de que uma relação pode ser definida em

termos de outras? Quando é impossível definir uma relação em termos de outras (e, portanto, deve-se adicioná-la aos primitivos da linguagem)?

8.2 Expressando Propriedades de Estruturas

Frequentemente é útil e importante expressar condições sobre funções e relações, ou, mais geralmente, que as funções e relações em uma estrutura satisfaçam essas condições. Por exemplo, nós gostaríamos de ter maneiras de distinguir aquelas estruturas para uma linguagem que "capturam" o que queremos que os predicados "signifiquem" daquelas que não capturam. É claro que somos completamente livres para especificar quais estruturas "pretendemos", por exemplo, podemos especificar que a interpretação do predicado $\leq$ deve ser uma ordenação, ou que estamos interessados apenas em interpretações de $\mathscr{L}$ nas quais o domínio consiste em conjuntos e $\in$ é interpretado pela relação "é um elemento de". Mas podemos fazer isso com sentenças da linguagem? Em outras palavras, quais condições sobre uma estrutura M podemos expressar por uma sentença (ou talvez um conjunto de sentenças) na linguagem de M? Há algumas condições que não conseguiremos expressar. Por exemplo, não há sentença de $\mathscr{L}_A$ que seja verdadeira em uma estrutura M somente se $|M| = \mathbb{N}$. Não podemos expressar "o domínio contém apenas números naturais". Mas há "propriedades estruturais" de estruturas que talvez possamos expressar. Quais propriedades de estruturas podemos expressar por sentenças? Ou, dito de outra forma, quais coleções de estruturas podemos descrever como aquelas que tornam uma sentença (ou conjunto de sentenças) verdadeira?

Definição 8.2 (Modelo de um conjunto). Seja Γ um conjunto de sentenças em uma linguagem $\mathscr{L}$. Nós dizemos que uma estrutura M *é um modelo de* Γ se $M \vDash A$ para todo $A \in \Gamma$.

Exemplo 8.3. A sentença $\forall x\, x \leq x$ é verdadeira em $\boldsymbol{M}$ sse $\leq^{M}$ é uma relação reflexiva. A sentença $\forall x \forall y\, ((x \leq y \wedge y \leq x) \rightarrow x = y)$ é verdadeira em $\boldsymbol{M}$ sse $\leq^{M}$ é antissimétrica. A sentença $\forall x \forall y \forall z\, ((x \leq y \wedge y \leq z) \rightarrow x \leq z)$ é verdadeira em $\boldsymbol{M}$ sse $\leq^{M}$ é transitiva. Assim, os modelos de

$$\begin{aligned} \{ \quad & \forall x\, x \leq x, \\ & \forall x \forall y\, ((x \leq y \wedge y \leq x) \rightarrow x = y), \\ & \forall x \forall y \forall z\, ((x \leq y \wedge y \leq z) \rightarrow x \leq z) \quad \} \end{aligned}$$

são exatamente aquelas estruturas nas quais $\leq^{M}$ é reflexiva, antissimétrica e transitiva, isto é, uma ordem parcial. Portanto, podemos tomá-las como axiomas para a *teoria de primeira ordem das ordens parciais.*

8.3 Exemplos de Teorias de Primeira Ordem

Exemplo 8.4. A teoria das ordens lineares estritas na linguagem $\mathcal{L}_{<}$ é axiomatizada pelo conjunto

$$\begin{aligned} \{ \quad & \forall x \neg x < x, \\ & \forall x \forall y\, ((x < y \vee y < x) \vee x = y), \\ & \forall x \forall y \forall z\, ((x < y \wedge y < z) \rightarrow x < z) \quad \} \end{aligned}$$

Ela captura completamente as estruturas pretendidas: toda ordem linear estrita é um modelo deste sistema de axiomas e, vice-versa, se R é uma ordem linear sobre um conjunto X, então a estrutura $\boldsymbol{M}$ com $|M| = X$ e $<^{M} = R$ é um modelo desta teoria.

Exemplo 8.5. A teoria dos grupos na linguagem $\mathtt{1}$ (símbolo de constante), $\cdot$ (símbolo de função de dois lugares) é axiomatizada por

$$\begin{aligned} & \forall x\, (x \cdot \mathtt{1}) = x \\ & \forall x \forall y \forall z\, (x \cdot (y \cdot z)) = ((x \cdot y) \cdot z) \\ & \forall x \exists y\, (x \cdot y) = \mathtt{1} \end{aligned}$$

Exemplo 8.6. A teoria da aritmética de Peano é axiomatizada pelas seguintes sentenças na linguagem da aritmética $\mathcal{L}_A$.

$$\forall x \forall y \, (x' = y' \to x = y)$$
$$\forall x \, 0 \neq x'$$
$$\forall x \, (x + 0) = x$$
$$\forall x \forall y \, (x + y') = (x + y)'$$
$$\forall x \, (x \times 0) = 0$$
$$\forall x \forall y \, (x \times y') = ((x \times y) + x)$$
$$\forall x \forall y \, (x < y \leftrightarrow \exists z \, (z' + x) = y)$$

mais todas as sentenças da forma

$$(A(0) \wedge \forall x \, (A(x) \to A(x'))) \to \forall x \, A(x)$$

Como há infinitas sentenças desta última forma, este sistema de axiomas é infinito. Esta última forma é chamada de *esquema de indução*. (Na verdade, o esquema de indução é um pouco mais complicado do que deixamos transparecer aqui.)

O último axioma é uma *definição explícita* de $<$.

Exemplo 8.7. A teoria dos conjuntos puros desempenha um papel importante nos fundamentos (e na filosofia) da matemática. Um conjunto é puro se todos os seus elementos são também conjuntos puros. O conjunto vazio conta, portanto, como puro, mas um conjunto que tem como um elemento algo que não é um conjunto não seria puro. Então os conjuntos puros são aqueles que são formados apenas a partir do conjunto vazio e sem "urelementos", isto é, objetos que não são, eles próprios, conjuntos.

O seguinte poderia ser considerado um sistema de axiomas para uma teoria dos conjuntos puros:

$$\exists x \neg \exists y \, y \in x$$
$$\forall x \forall y \, (\forall z (z \in x \leftrightarrow z \in y) \to x = y)$$
$$\forall x \forall y \exists z \forall u \, (u \in z \leftrightarrow (u = x \vee u = y))$$
$$\forall x \exists y \forall z \, (z \in y \leftrightarrow \exists u \, (z \in u \wedge u \in x))$$

mais todas as sentenças da forma

$$\exists x \, \forall y \, (y \in x \leftrightarrow A(y))$$

O primeiro axioma diz que há um conjunto sem elementos (isto é, $\emptyset$ existe); o segundo diz que conjuntos são extensionais; o terceiro que, para quaisquer conjuntos X e Y, o conjunto $\{X, Y\}$ existe; o quarto que, para qualquer conjunto X, o conjunto $\cup X$ existe, onde $\cup X$ é a união de todos os elementos de X.

As sentenças mencionadas por último são chamadas coletivamente de *esquema de compreensão ingênuo*. Ele essencialmente diz que, para todo $A(x)$, o conjunto $\{x : A(x)\}$ existe—então, à primeira vista, um axioma verdadeiro, útil e talvez até necessário. É chamado de "ingênuo" porque, como se vê, ele torna esta teoria insatisfatível: se você toma $A(y)$ como $\neg y \in y$, obtém a sentença

$$\exists x \, \forall y \, (y \in x \leftrightarrow \neg y \in y)$$

e esta sentença não é satisfeita em nenhuma estrutura.

Exemplo 8.8. Na área da *mereologia*, a relação de *parte* é uma relação fundamental. Assim como as teorias de conjuntos, há teorias de parte que axiomatizam várias concepções (às vezes conflitantes) dessa relação.

A linguagem da mereologia contém um único símbolo de predicado de dois lugares P, e $P(x, y)$ "significa" que x é uma parte de y. Quando temos essa interpretação em mente, uma estrutura para essa linguagem é chamada de *estrutura de parte*. É claro que nem toda estrutura para um único predicado de dois lugares realmente merecerá esse nome. Para ter uma chance de capturar "parte", P^M deve satisfazer algumas condições, que podemos estabelecer como axiomas para uma teoria de parte. Por exemplo, a parte é uma ordem parcial sobre objetos: todo objeto é uma parte (ainda que uma parte *imprópria*) de si mesmo; dois objetos diferentes não podem ser partes um do outro; uma parte de uma parte de um objeto é, ela própria, parte daquele objeto. Note

que, nesse sentido, "é uma parte de" se assemelha a "é um subconjunto de", mas não se assemelha a "é um elemento de", que não é nem reflexiva nem transitiva.

$$\forall x\, P(x,x)$$
$$\forall x\, \forall y\, ((P(x,y) \wedge P(y,x)) \rightarrow x = y)$$
$$\forall x\, \forall y\, \forall z\, ((P(x,y) \wedge P(y,z)) \rightarrow P(x,z))$$

Além disso, quaisquer dois objetos têm uma soma mereológica (um objeto que tem esses dois objetos como partes, e é minimal nesse aspecto).

$$\forall x\, \forall y\, \exists z\, \forall u\, (P(z,u) \leftrightarrow (P(x,u) \wedge P(y,u)))$$

Estes são apenas alguns dos princípios básicos de parte considerados pelos metafísicos. Princípios adicionais, no entanto, rapidamente se tornam difíceis de formular ou escrever sem primeiro introduzir algumas relações definidas. Por exemplo, a maioria dos metafísicos interessados em mereologia também considera o seguinte como um princípio válido: sempre que um objeto x tem uma parte própria y, ele também tem uma parte z que não tem partes em comum com y, e tal que a fusão de y e z é x.

8.4 Expressando Relações em uma Estrutura

Um uso principal que se pode dar às fórmulas é expressar propriedades e relações em uma estrutura $\mathfrak{M}$ em termos dos primitivos da linguagem $\mathcal{L}$ de $\mathfrak{M}$. Com isso queremos dizer o seguinte: o domínio de $\mathfrak{M}$ é um conjunto de objetos. Os símbolos de constante, símbolos de função e predicados são interpretados em $\mathfrak{M}$ por alguns objetos em $|\mathfrak{M}|$, funções sobre $|\mathfrak{M}|$, e relações sobre $|\mathfrak{M}|$. Por exemplo, se A^2_0 está em $\mathcal{L}$, então $\mathfrak{M}$ atribui a ele uma relação $R = A^{2\mathfrak{M}}_0$. Então a fórmula $A^2_0(v_1, v_2)$ *expressa* exatamente

essa relação, no seguinte sentido: se uma atribuição de variáveis s mapeia v_1 para $a \in |M|$ e v_2 para $b \in |M|$, então

$$Rab \quad \text{sse} \quad M, s \vDash A^2_0(v_1, v_2).$$

Note que temos que envolver atribuições de variáveis aqui: não podemos simplesmente dizer "Rab sse $M \vDash A^2_0(a, b)$" porque a e b não são símbolos da nossa linguagem: eles são elementos de $|M|$.

Como não temos apenas fórmulas atômicas, mas podemos combiná-las usando os conectivos lógicos e os quantificadores, fórmulas mais complexas podem definir outras relações que não estão diretamente embutidas em M. Estamos interessados em como fazer isso e, especificamente, quais relações podemos definir em uma estrutura.

Definição 8.9. Seja $A(v_1, \ldots, v_n)$ uma fórmula de $\mathcal{L}$ na qual apenas $v_1, \ldots, v_n$ ocorrem livres, e seja M uma estrutura para $\mathcal{L}$. $A(v_1, \ldots, v_n)$ *expressa a relação* $R \subseteq |M|^n$ sse

$$Ra_1 \ldots a_n \quad \text{sse} \quad M, s \vDash A(v_1, \ldots, v_n)$$

para qualquer atribuição de variáveis s com $s(v_i) = a_i$ $(i = 1, \ldots, n)$.

Exemplo 8.10. No modelo padrão da aritmética N, a fórmula $v_1 < v_2 \vee v_1 = v_2$ expressa a relação $\leq$ sobre $\mathbb{N}$. A fórmula $v_2 = v_1'$ expressa a relação sucessora, isto é, a relação $R \subseteq \mathbb{N}^2$ onde Rnm vale se m é o sucessor de n. A fórmula $v_1 = v_2'$ expressa a relação predecessora. As fórmulas $\exists v_3\,(v_3 \neq 0 \wedge v_2 = (v_1 + v_3))$ e $\exists v_3\,(v_1 + v_3') = v_2$ ambas expressam a relação $<$. Isso significa que o símbolo de predicado $<$ é, na verdade, supérfluo na linguagem da aritmética; ele pode ser definido.

Essa ideia não é interessante apenas em estruturas específicas, mas geralmente sempre que usamos uma linguagem para descrever um modelo ou modelos pretendidos, isto é, quando

consideramos teorias. Essas teorias frequentemente contêm apenas alguns predicados como símbolos básicos, mas no domínio que elas são usadas para descrever muitas outras relações frequentemente desempenham um papel importante. Se essas outras relações podem ser sistematicamente expressas pelas relações que interpretam os predicados básicos da linguagem, nós dizemos que podemos *defini-las* na linguagem.

8.5 A Teoria dos Conjuntos

Quase toda a matemática pode ser desenvolvida na teoria dos conjuntos. Desenvolver a matemática nessa teoria envolve uma série de coisas. Primeiro, requer um conjunto de axiomas para a relação $\in$. Vários sistemas de axiomas diferentes foram desenvolvidos, às vezes com propriedades conflitantes de $\in$. O sistema de axiomas conhecido como **ZFC**, a teoria dos conjuntos de Zermelo–Fraenkel com o axioma da escolha, destaca-se: é de longe o mais amplamente usado e estudado, porque acontece que seus axiomas bastam para provar quase todas as coisas que os matemáticos esperam ser capazes de provar. Mas antes que isso possa ser estabelecido, é primeiro necessário deixar claro como podemos sequer *expressar* todas as coisas que os matemáticos gostariam de expressar. Para começar, a linguagem não contém símbolos de constante nem símbolos de função, então parece, à primeira vista, pouco claro que possamos falar sobre conjuntos particulares (como $\emptyset$ ou $\mathbb{N}$), falar sobre operações sobre conjuntos (como $X \cup Y$ e $\wp(X)$), muito menos outras construções que envolvem coisas diferentes de conjuntos, como relações e funções.

Para começar, "é um elemento de" não é a única relação na qual estamos interessados: "é um subconjunto de" parece quase tão importante. Mas nós podemos *definir* "é um subconjunto de" em termos de "é um elemento de". Para fazer isso, temos que encontrar uma fórmula $A(x,y)$ na linguagem da teoria dos conjuntos que é satisfeita por um par de conjuntos $\langle X, Y \rangle$ sse

$X \subseteq Y$. Mas X é um subconjunto de Y exatamente quando todos os elementos de X são também elementos de Y. Então podemos definir $\subseteq$ pela fórmula

$$\forall z\,(z \in x \to z \in y)$$

Agora, sempre que quisermos usar a relação $\subseteq$ em uma fórmula, nós poderíamos, em vez disso, usar aquela fórmula (com x e y adequadamente substituídos, e a variável ligada z renomeada se necessário). Por exemplo, a extensionalidade dos conjuntos significa que se quaisquer conjuntos x e y estão contidos um no outro, então x e y devem ser o mesmo conjunto. Isso pode ser expresso por $\forall x\,\forall y\,((x \subseteq y \wedge y \subseteq x) \to x = y)$, ou, se substituirmos $\subseteq$ pela definição acima, por

$$\forall x\,\forall y\,((\forall z\,(z \in x \to z \in y) \wedge \forall z\,(z \in y \to z \in x)) \to x = y).$$

Este é, de fato, um dos axiomas de **ZFC**, o "axioma da extensionalidade".

Não há símbolo de constante para $\emptyset$, mas podemos expressar "x é vazio" por $\neg\exists y\, y \in x$. Então "$\emptyset$ existe" torna-se a sentença $\exists x\,\neg\exists y\, y \in x$. Este é outro axioma de **ZFC**. (Note que o axioma da extensionalidade implica que há apenas um conjunto vazio.) Sempre que quisermos falar sobre $\emptyset$ na linguagem da teoria dos conjuntos, nós escreveríamos isso como "há um conjunto que é vazio e ..." Como exemplo, para expressar o fato de que $\emptyset$ é um subconjunto de todo conjunto, poderíamos escrever

$$\exists x\,(\neg\exists y\, y \in x \wedge \forall z\, x \subseteq z)$$

onde, é claro, $x \subseteq z$ teria, por sua vez, que ser substituído por sua definição.

Para falar sobre operações sobre conjuntos, como $X \cup Y$ e $\wp(X)$, nós temos que usar um truque semelhante. Não há símbolos de função na linguagem da teoria dos conjuntos, mas podemos expressar as relações funcionais $X \cup Y = Z$ e $\wp(X) = Y$

por

$$\forall u\,((u \in x \vee u \in y) \leftrightarrow u \in z)$$
$$\forall u\,(u \subseteq x \leftrightarrow u \in y)$$

já que os elementos de $X \cup Y$ são exatamente os conjuntos que são ou elementos de X ou elementos de Y, e os elementos de $\wp(X)$ são exatamente os subconjuntos de X. No entanto, isso não nos permite usar $x \cup y$ ou $\wp(x)$ como se fossem termos: só podemos usar as fórmulas inteiras que definem as relações $X \cup Y = Z$ e $\wp(X) = Y$. De fato, não sabemos que essas relações são alguma vez satisfeitas, isto é, não sabemos que uniões e conjuntos potência sempre existem. Por exemplo, a sentença $\forall x\,\exists y\,\wp(x) = y$ é outro axioma de **ZFC** (o axioma do conjunto potência).

E quanto a falar de pares ordenados ou funções? Aqui temos que explicar como podemos pensar em pares ordenados e funções como tipos especiais de conjuntos. Uma maneira de definir o par ordenado $\langle x,y\rangle$ é como o conjunto $\{\{x\},\{x,y\}\}$. Mas, como antes, não podemos introduzir um símbolo de função que nomeie esse conjunto; só podemos definir a relação $\langle x,y\rangle = z$, isto é, $\{\{x\},\{x,y\}\} = z$:

$$\forall u\,(u \in z \leftrightarrow (\forall v\,(v \in u \leftrightarrow v = x) \vee \forall v\,(v \in u \leftrightarrow (v = x \vee v = y))))$$

Isso diz que os elementos u de z são exatamente aqueles conjuntos que ou têm x como seu único elemento ou têm x e y como seus únicos elementos (em outras palavras, aqueles conjuntos que são ou idênticos a $\{x\}$ ou idênticos a $\{x,y\}$). Uma vez que temos isso, podemos dizer outras coisas, por exemplo, que $X \times Y = Z$:

$$\forall z\,(z \in Z \leftrightarrow \exists x\,\exists y\,(x \in X \wedge y \in Y \wedge \langle x,y\rangle = z))$$

Uma função $f\colon X \to Y$ pode ser pensada como a relação $f(x) = y$, isto é, como o conjunto de pares $\{\langle x,y\rangle : f(x) = y\}$. Nós podemos então dizer que um conjunto f é uma função de X em Y se (a) é uma relação $\subseteq X \times Y$, (b) é total, isto é, para todo $x \in X$ há algum $y \in Y$ tal que $\langle x,y\rangle \in f$ e (c) é funcional,

isto é, sempre que $\langle x,y\rangle, \langle x,y'\rangle \in f$, $y = y'$ (porque os valores de funções devem ser únicos). Então "f é uma função de X em Y" pode ser escrito como:

$$\forall u\,(u \in f \to \exists x\,\exists y\,(x \in X \wedge y \in Y \wedge \langle x,y\rangle = u)) \wedge$$
$$\forall x\,(x \in X \to (\exists y\,(y \in Y \wedge \text{mapeia}(f,x,y)) \wedge$$
$$(\forall y\,\forall y'\,((\text{mapeia}(f,x,y) \wedge \text{mapeia}(f,x,y')) \to y = y')))$$

onde $\text{mapeia}(f,x,y)$ abrevia $\exists v\,(v \in f \wedge \langle x,y\rangle = v)$ (esta fórmula expressa "$f(x) = y$").

Agora também não é difícil expressar que $f\colon X \to Y$ é injetora, por exemplo:

$$f\colon X \to Y \wedge \forall x\,\forall x'\,((x \in X \wedge x' \in X \wedge$$
$$\exists y\,(\text{mapeia}(f,x,y) \wedge \text{mapeia}(f,x',y))) \to x = x')$$

Uma função $f\colon X \to Y$ é injetora sse, sempre que f mapeia $x,x' \in X$ para um único y, $x = x'$. Se abreviarmos esta fórmula como $\text{inj}(f,X,Y)$, já estamos em posição de afirmar na linguagem da teoria dos conjuntos algo tão não trivial quanto o teorema de Cantor: não há função injetora de $\wp(X)$ em X:

$$\forall X\,\forall Y\,(\wp(X) = Y \to \neg\exists f\,\text{inj}(f,Y,X))$$

Poderíamos pensar que a teoria dos conjuntos requer outro axioma que garanta a existência de um conjunto para toda propriedade definidora. Se $A(x)$ é uma fórmula da teoria dos conjuntos com a variável x livre, podemos considerar a sentença

$$\exists y\,\forall x\,(x \in y \leftrightarrow A(x)).$$

Esta sentença afirma que há um conjunto y cujos elementos são todos e somente aqueles x que satisfazem $A(x)$. Este esquema é chamado de "princípio da compreensão". Parece muito útil; infelizmente é inconsistente. Tome $A(x) \equiv \neg x \in x$, então o princípio da compreensão afirma

$$\exists y\,\forall x\,(x \in y \leftrightarrow x \notin x),$$

isto é, afirma a existência de um conjunto de todos os conjuntos que não são elementos de si mesmos. Nenhum tal conjunto pode existir—este é o Paradoxo de Russell. **ZFC**, de fato, contém uma versão restrita—e consistente—deste princípio, o princípio da separação:

$$\forall z\, \exists y\, \forall x\, (x \in y \leftrightarrow (x \in z \wedge A(x))).$$

8.6 Expressando o Tamanho de Estruturas

Há algumas propriedades de estruturas que podemos expressar mesmo sem usar os símbolos não lógicos de uma linguagem. Por exemplo, há sentenças que são verdadeiras em uma estrutura se, e somente se, o domínio da estrutura tem pelo menos, no máximo ou exatamente um certo número n de elementos.

Proposição 8.11. *A sentença*

$$\begin{aligned} A_{\geq n} \equiv \exists x_1\, \exists x_2\, \ldots \exists x_n \\ (x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge x_1 \neq x_4 \wedge \cdots \wedge x_1 \neq x_n \wedge \\ x_2 \neq x_3 \wedge x_2 \neq x_4 \wedge \cdots \wedge x_2 \neq x_n \wedge \\ \vdots \\ x_{n-1} \neq x_n) \end{aligned}$$

é verdadeira em uma estrutura $\mathbf{M}$ se, e somente se, $|\mathbf{M}|$ contém pelo menos n elementos. Consequentemente, $\mathbf{M} \vDash \neg A_{\geq n+1}$ se, e somente se, $|\mathbf{M}|$ contém no máximo n elementos.

Proposição 8.12. *A sentença*

$$\begin{aligned} A_{=n} \equiv \exists x_1 \exists x_2 \ldots \exists x_n \\ (x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge x_1 \neq x_4 \wedge \cdots \wedge x_1 \neq x_n \wedge \\ x_2 \neq x_3 \wedge x_2 \neq x_4 \wedge \cdots \wedge x_2 \neq x_n \wedge \\ \vdots \\ x_{n-1} \neq x_n \wedge \\ \forall y\, (y = x_1 \vee \cdots \vee y = x_n)) \end{aligned}$$

é verdadeira em uma estrutura $\mathbf{M}$ se, e somente se, $|M|$ contém exatamente n elementos.

Proposição 8.13. *Uma estrutura é infinita se, e somente se, é um modelo de*

$$\{A_{\geq 1}, A_{\geq 2}, A_{\geq 3}, \ldots\}.$$

Não há uma única sentença puramente lógica que seja verdadeira em M se, e somente se, $|M|$ é infinito. No entanto, pode-se dar sentenças com predicados não lógicos que só têm modelos infinitos (embora nem toda estrutura infinita seja modelo delas). A propriedade de ser uma estrutura finita e a propriedade de ser uma estrutura incontável nem sequer podem ser expressas com um conjunto infinito de sentenças. Estes fatos seguem dos teoremas da compacidade e de Löwenheim–Skolem.

Resumo

Os conjuntos de sentenças, em certo sentido, descrevem as estruturas nas quais são conjuntamente verdadeiras; essas estruturas são seus **modelos**. Reciprocamente, se começamos com uma estrutura ou um conjunto de estruturas, podemos estar interessados no conjunto de sentenças das quais elas são modelos; esta é a **teoria** da estrutura ou do conjunto de estruturas. Qualquer

conjunto desse tipo de sentenças tem a propriedade de que toda sentença que é consequência delas já está no conjunto; elas são **fechadas**. De modo mais geral, chamamos um conjunto Γ de teoria se ele é fechado sob consequência, e dizemos que Γ é **axiomatizado** por Δ se Γ consiste em todas as sentenças que são consequência de Δ.

A matemática fornece muitos exemplos de teorias, por exemplo, as teorias das ordens lineares, dos grupos, ou teorias da aritmética, por exemplo, a teoria axiomatizada pelos axiomas de Peano. Mas há muitos exemplos de teorias importantes em outras disciplinas também, por exemplo, os bancos de dados relacionais podem ser pensados como teorias, e a metafísica ocupa-se de teorias da relação parte-todo, que podem ser axiomatizadas.

Uma questão significativa ao montar uma teoria para estudo é se sua linguagem é expressiva o bastante para nos permitir formular tudo o que queremos que a teoria trate, e outra é se ela é forte o bastante para provar o que queremos que ela prove. Para **expressar** uma relação, precisamos de uma fórmula com o número necessário de variáveis livres. Na **teoria dos conjuntos**, temos apenas $\in$ como símbolo de relação, mas ele nos permite expressar $x \subseteq y$ usando $\forall u\,(u \in x \to u \in y)$. A **teoria dos conjuntos de Zermelo-Fraenkel ZFC**, de fato, é forte o bastante tanto para expressar (quase) toda afirmação matemática quanto para (quase) provar todo teorema matemático usando um punhado de axiomas e uma cadeia de definições cada vez mais complicadas, como a de $\subseteq$.

Problemas

Problema 8.1. Encontre fórmulas em $\mathcal{L}_A$ que definam as seguintes relações:

1. n está entre i e j;
2. n divide m exatamente (isto é, m é um múltiplo de n);

3. n é um número primo (isto é, nenhum número além de 1 e n divide n exatamente).

Problema 8.2. Suponha que a fórmula $A(v_1, v_2)$ expressa a relação $R \subseteq |M|^2$ em uma estrutura M. Encontre fórmulas que expressem as seguintes relações:

1. a inversa R^{-1} de R;
2. o produto relativo $R \mid R$;

Você consegue encontrar uma maneira de expressar R^+, o fecho transitivo de R?

Problema 8.3. Seja $\mathcal{L}$ a linguagem contendo apenas um símbolo de predicado de 2 lugares $<$ (sem outros símbolos de constante, símbolos de função ou predicados— exceto, é claro, $=$). Seja N a estrutura tal que $|N| = \mathbb{N}$, e $<^N = \{\langle n, m\rangle : n < m\}$. Prove o seguinte:

1. $\{0\}$ é definível em N;
2. $\{1\}$ é definível em N;
3. $\{2\}$ é definível em N;
4. para cada $n \in \mathbb{N}$, o conjunto $\{n\}$ é definível em N;
5. todo subconjunto finito de $|N|$ é definível em N;
6. todo subconjunto cofinito de $|N|$ é definível em N (onde $X \subseteq \mathbb{N}$ é cofinito sse $\mathbb{N} \setminus X$ é finito).

Problema 8.4. Mostre que o princípio da compreensão é inconsistente dando uma derivação que mostre

$$\exists y \, \forall x \, (x \in y \leftrightarrow x \notin x) \vdash \bot.$$

Pode ajudar mostrar primeiro $(A \to \neg A) \wedge (\neg A \to A) \vdash \bot$.

CAPÍTULO 9

Sistemas de Derivação

9.1 Introdução

As lógicas comumente têm tanto uma semântica quanto um sistema de derivação. A semântica trata de conceitos como verdade, satisfatibilidade, validade e consequência. O propósito dos sistemas de derivação é fornecer um método puramente sintático de estabelecer consequência e validade. Eles são puramente sintáticos no sentido de que uma derivação em tal sistema é um objeto sintático finito, geralmente uma sequência (ou outra disposição finita) de sentenças ou fórmulas. Bons sistemas de derivação têm a propriedade de que qualquer sequência ou disposição dada de sentenças ou fórmulas pode ser verificada mecanicamente como "correta".

Os sistemas de derivação mais simples (e historicamente os primeiros) para a lógica de primeira ordem foram *axiomáticos*. Uma sequência de fórmulas conta como uma derivação em tal sistema se cada fórmula individual nela é ou está entre um conjunto fixo de "axiomas" ou segue de fórmulas anteriores na sequência por uma entre um número fixo de "regras de inferência"—e pode ser verificado mecanicamente se uma fórmula é um axioma

e se segue corretamente de outras fórmulas por uma das regras de inferência. Sistemas de derivação axiomáticos são fáceis de descrever—e também fáceis de tratar meta-teoricamente—mas as derivações neles são difíceis de ler e entender, e também difíceis de produzir.

Outros sistemas de derivação foram desenvolvidos com o objetivo de facilitar a construção de derivações ou facilitar a compreensão de derivações uma vez completas. Exemplos são a dedução natural, árvores de verdade, também conhecidas como provas por tableaux, e o cálculo de sequentes. Alguns sistemas de derivação são projetados especialmente com a mecanização em mente; por exemplo, o método da resolução é fácil de implementar em software (mas suas derivações são essencialmente impossíveis de entender). A maioria desses outros sistemas de derivação representa derivações como árvores de fórmulas em vez de sequências. Isso facilita ver quais partes de uma derivação dependem de quais outras partes.

Assim, para uma dada lógica, como a lógica de primeira ordem, os diferentes sistemas de derivação darão diferentes explicações do que significa uma sentença ser um *teorema* e o que significa uma sentença ser derivável a partir de outras. Seja como for feito isso (via derivações axiomáticas, deduções naturais, derivações de sequentes, árvores de verdade, refutações por resolução), queremos que essas relações correspondam às noções semânticas de validade e consequência. Escrevamos $\vdash A$ para "A é um teorema" e "$\Gamma \vdash A$" para "A é derivável de Γ." Seja como for $\vdash$ definido, queremos que corresponda a $\models$, isto é:

1. $\vdash A$ se, e somente se, $\models A$

2. $\Gamma \vdash A$ se, e somente se, $\Gamma \models A$

A direção "somente se" acima é chamada de *correção*. Um sistema de derivação é correto se a derivabilidade garante consequência (ou validade). Todo sistema de derivação decente tem que ser correto; sistemas de derivação incorretos não são úteis de modo algum. Afinal, o propósito inteiro de uma derivação é fornecer

uma garantia sintática de validade ou consequência. Provaremos a correção para os sistemas de derivação que apresentamos.

A direção conversa "se" também é importante: é chamada de *completude*. Um sistema de derivação completo é forte o suficiente para mostrar que A é um teorema sempre que A é válida, e que $\Gamma \vdash A$ sempre que $\Gamma \models A$. A completude é mais difícil de estabelecer, e algumas lógicas não têm sistemas de derivação completos. A lógica de primeira ordem tem. Kurt Gödel foi o primeiro a provar a completude para um sistema de derivação da lógica de primeira ordem em sua dissertação de 1929.

Outro conceito ligado aos sistemas de derivação é o de *consistência*. Um conjunto de sentenças é chamado inconsistente se qualquer coisa pode ser derivada dele, e consistente caso contrário. A inconsistência é a contrapartida sintática da insatisfatibilidade: como conjuntos insatisfatíveis, conjuntos inconsistentes de sentenças não fazem boas teorias; são defeituosos de maneira fundamental. Conjuntos consistentes de sentenças podem não ser verdadeiros ou úteis, mas pelo menos passam esse limiar mínimo de utilidade lógica. Para diferentes sistemas de derivação, a definição específica de consistência de conjuntos de sentenças pode diferir, mas como $\vdash$, queremos que a consistência coincida com sua contrapartida semântica, a satisfatibilidade. Queremos que seja sempre o caso que Γ é consistente se, e somente se, é satisfatível. Aqui, a direção "somente se" corresponde à completude (consistência garante satisfatibilidade), e a direção "se" corresponde à correção (satisfatibilidade garante consistência). De fato, para a lógica clássica de primeira ordem, as duas versões de correção e completude são equivalentes.

9.2 O Cálculo de Sequentes

Enquanto muitos sistemas de derivação operam com disposições de sentenças, o cálculo de sequentes opera com *sequentes*. Um sequente é uma expressão da forma

$$A_1, \ldots, A_m \Rightarrow B_1, \ldots, B_m,$$

isto é, um par de sequências de sentenças, separadas pelo símbolo de sequente $\Rightarrow$. Qualquer uma das sequências pode ser vazia. Uma derivação no cálculo de sequentes é uma árvore de sequentes, em que os sequentes mais altos são de uma forma especial (são chamados de "sequentes iniciais" ou "axiomas") e todo outro sequente segue dos sequentes imediatamente acima dele por uma das regras de inferência. As regras de inferência ou manipulam as sentenças nos sequentes (adicionando, removendo ou rearranjando-as à esquerda ou à direita), ou introduzem uma fórmula complexa na conclusão da regra. Por exemplo, a regra $\wedge$L permite a inferência de $A, \Gamma \Rightarrow \Delta$ para $A \wedge B, \Gamma \Rightarrow \Delta$, e a $\rightarrow$R permite a inferência de $A, \Gamma \Rightarrow \Delta, B$ para $\Gamma \Rightarrow \Delta, A \rightarrow B$, para quaisquer Γ, Δ, A e B. (Em particular, Γ e Δ podem ser vazios.)

A relação $\vdash$ baseada no cálculo de sequentes é definida da seguinte forma: $\Gamma \vdash A$ se, e somente se, há alguma sequência Γ_0 tal que toda A em Γ_0 está em Γ e há uma derivação com o sequente $\Gamma_0 \Rightarrow A$ em sua raiz. A é um teorema no cálculo de sequentes se o sequente $\Rightarrow A$ tem uma derivação. Por exemplo, aqui está uma derivação que mostra que $\vdash (A \wedge B) \rightarrow A$:

$$\dfrac{\dfrac{A \Rightarrow A}{A \wedge B \Rightarrow A}\,\wedge\text{L}}{\Rightarrow (A \wedge B) \rightarrow A}\,\rightarrow\text{R}$$

Um conjunto Γ é inconsistente no cálculo de sequentes se há uma derivação de $\Gamma_0 \Rightarrow$ (em que toda $A \in \Gamma_0$ está em Γ e o lado direito do sequente é vazio). Usando a regra WR, qualquer sentença pode ser derivada a partir de um conjunto inconsistente.

O cálculo de sequentes foi inventado na década de 1930 por Gerhard Gentzen. Por causa de seu desenho sistemático e simétrico, é um formalismo muito útil para desenvolver uma teoria de derivações. É relativamente fácil encontrar derivações no cálculo de sequentes, mas essas derivações são frequentemente difíceis de ler e sua conexão com provas às vezes não é fácil de ver. Provou ser uma abordagem muito elegante a sistemas de derivação, no entanto, e muitas lógicas têm sistemas de cálculo de sequentes.

9.3 Dedução Natural

A dedução natural é um sistema de derivação destinado a espelhar o raciocínio real (especialmente o tipo de raciocínio regimentado empregado por matemáticos). O raciocínio real procede por vários padrões "naturais". Por exemplo, a prova por casos nos permite estabelecer uma conclusão com base em uma premissa disjuntiva, estabelecendo que a conclusão segue de qualquer um dos disjuntos. A prova indireta nos permite estabelecer uma conclusão mostrando que sua negação leva a uma contradição. A prova condicional estabelece uma afirmação condicional "se ... então ..." mostrando que o consequente segue do antecedente. A dedução natural é uma formalização de algumas dessas inferências naturais. Cada um dos conectivos lógicos e quantificadores vem com duas regras, uma de introdução e uma de eliminação, e cada uma corresponde a um desses padrões de inferência natural. Por exemplo, $\rightarrow$Intro corresponde à prova condicional, e $\vee$Elim à prova por casos. Uma regra particularmente simples é $\wedge$Elim, que permite a inferência de $A \wedge B$ para A (ou B).

Uma característica que distingue a dedução natural de outros sistemas de derivação é o uso de hipóteses. Uma derivação em dedução natural é uma árvore de fórmulas. Uma única fórmula fica na raiz da árvore de fórmulas, e as "folhas" da árvore são fórmulas das quais a conclusão é derivada. Na dedução natural, algumas fórmulas folha desempenham um papel dentro da derivação, mas são "consumidas" quando a derivação atinge a conclusão. Isso corresponde à prática, no raciocínio real, de introduzir hipóteses que só permanecem em vigor por um curto período. Por exemplo, em uma prova por casos, assumimos a verdade de cada um dos disjuntos; na prova condicional, assumimos a verdade do antecedente; na prova indireta, assumimos a verdade da negação da conclusão. Essa maneira de introduzir hipóteses hipotéticas e depois descartá-las a serviço de estabelecer um passo intermediário é uma marca da dedução natural. As fórmulas nas folhas de uma derivação em dedução natural são cha-

madas de hipóteses, e algumas das regras de inferência podem descarregá-las. Por exemplo, se temos uma derivação de B a partir de algumas hipóteses que incluem A, então a regra $\rightarrow$Intro nos permite inferir $A \rightarrow B$ e descartar qualquer hipótese da forma A. (Para acompanhar quais hipóteses são descartadas em quais inferências, rotulamos a inferência e as hipóteses que ela descarta com um número.) As hipóteses que permanecem não descarregadas no final da derivação são, juntas, suficientes para a verdade da conclusão, de modo que uma derivação estabelece que suas hipóteses não descarregadas implicam sua conclusão.

A relação $\Gamma \vdash A$ baseada na dedução natural vale se, e somente se, há uma derivação em que A é a última sentença na árvore, e toda folha que está não descarregada está em Γ. A é um teorema na dedução natural se, e somente se, há uma derivação em que A é a última sentença e todas as hipóteses são descarregadas. Por exemplo, aqui está uma derivação que mostra que $\vdash (A \wedge B) \rightarrow A$:

$$\dfrac{\dfrac{[A \wedge B]^1}{A}\,\wedge\text{Elim}}{(A \wedge B) \rightarrow A}\,{}^{1}\,\rightarrow\text{Intro}$$

O rótulo 1 indica que a hipótese $A \wedge B$ é descarregada na inferência $\rightarrow$Intro.

Um conjunto Γ é inconsistente se, e somente se, $\Gamma \vdash \bot$ na dedução natural. A regra $\bot_I$ faz com que, a partir de um conjunto inconsistente, qualquer sentença possa ser derivada.

Sistemas de dedução natural foram desenvolvidos por Gerhard Gentzen e Stanisław Jaśkowski na década de 1930, e depois por Dag Prawitz e Frederic Fitch. Porque suas inferências espelham métodos naturais de prova, é favorecida por filósofos. As versões desenvolvidas por Fitch são frequentemente usadas em livros introdutórios de lógica. Na filosofia da lógica, as regras da dedução natural às vezes foram tomadas como dando os significados dos operadores lógicos ("semântica da Teoria da Prova").

9.4 Tableaux

Enquanto muitos sistemas de derivação operam com disposições de sentenças, os tableaux operam com fórmulas assinadas. Uma fórmula assinada é um par consistindo em um sinal de valor de verdade ($\mathbb{T}$ ou $\mathbb{F}$) e uma sentença

$$\mathbb{T}\,A \text{ ou } \mathbb{F}\,A.$$

Um tableau consiste em fórmulas assinadas dispostas em uma árvore ramificada para baixo. Começa com várias *hipóteses* e continua com fórmulas assinadas que resultam de uma das fórmulas assinadas acima dela pela aplicação de uma das regras de inferência. Cada regra nos permite adicionar uma ou mais fórmulas assinadas ao final de um ramo, ou duas fórmulas assinadas lado a lado—neste caso um ramo se divide em dois, com as duas fórmulas assinadas adicionadas formando os finais dos dois ramos.

Uma regra aplicada a uma fórmula assinada complexa resulta na adição de fórmulas assinadas que são sub-fórmulas imediatas. Elas vêm em pares, uma regra para cada um dos dois sinais. Por exemplo, a regra $\wedge\mathbb{T}$ se aplica a $\mathbb{T}\,A \wedge B$, e permite a adição tanto das duas fórmulas assinadas $\mathbb{T}\,A$ quanto de $\mathbb{T}\,B$ ao final de qualquer ramo que contém $\mathbb{T}\,A \wedge B$, e a regra $A \wedge B\mathbb{F}$ permite que um ramo seja dividido adicionando $\mathbb{F}\,A$ e $\mathbb{F}\,B$ lado a lado. Um tableau é fechado se cada um de seus ramos contém um par correspondente de fórmulas assinadas $\mathbb{T}\,A$ e $\mathbb{F}\,A$.

A relação $\vdash$ baseada em tableaux é definida da seguinte forma: $\Gamma \vdash A$ se, e somente se, há algum conjunto finito $\Gamma_0 = \{B_1, \ldots, B_n\} \subseteq \Gamma$ tal que há um tableau fechado para as hipóteses

$$\{\mathbb{F}\,A, \mathbb{T}\,B_1, \ldots, \mathbb{T}\,B_n\}$$

Por exemplo, aqui está um tableau fechado que mostra que $\vdash (A \wedge B) \to A$:

1.	$\mathbb{F}\,(A \wedge B) \to A$	Hipótese
2.	$\mathbb{T}\,A \wedge B$	$\to\mathbb{F}\,1$
3.	$\mathbb{F}\,A$	$\to\mathbb{F}\,1$
4.	$\mathbb{T}\,A$	$\to\mathbb{T}\,2$
5.	$\mathbb{T}\,B$	$\to\mathbb{T}\,2$
	$\otimes$	

Um conjunto Γ é inconsistente no cálculo de tableau se há um tableau fechado para hipóteses

$$\{\mathbb{T}\,B_1, \ldots, \mathbb{T}\,B_n\}$$

para algum $B_i \in \Gamma$.

Os Tableaux foram inventados na década de 1950 independentemente por Evert Beth e Jaakko Hintikka, e simplificados e popularizados por Raymond Smullyan. São muito fáceis de usar, pois construir um tableau é um procedimento muito sistemático. Por causa da natureza sistemática dos tableaux, eles também se prestam à implementação por computador. No entanto, um tableau é frequentemente difícil de ler e sua conexão com provas às vezes não é fácil de ver. A abordagem também é bastante geral, e muitas lógicas diferentes têm sistemas de tableau. Os Tableaux também nos ajudam a encontrar estruturas que satisfazem sentenças (ou conjuntos de sentenças) dadas: se o conjunto é satisfatível, ele não terá um tableau fechado, isto é, qualquer tableau terá um ramo aberto. A estrutura satisfatória pode ser "lida" de um ramo aberto, desde que toda regra que seja possível aplicar tenha sido aplicada naquele ramo. Há também uma conexão muito estreita com o cálculo de sequentes: essencialmente, um tableau fechado é uma derivação condensada no cálculo de sequentes, escrita de cabeça para baixo.

9.5 Dedução Axiomática

As derivações axiomáticas são os sistemas de derivação lógicos mais antigos e mais simples. Suas derivações são simplesmente

sequências de sentenças. Uma sequência de sentenças conta como uma derivação correta se toda sentença A nela satisfaz uma das seguintes condições:

1. A é um axioma, ou
2. A é um elemento de um dado conjunto Γ de sentenças, ou
3. A é justificada por uma regra de inferência.

Para ser um axioma, A tem que ter a forma de um entre vários esquemas fixos de sentença. Há muitos conjuntos de esquemas de axiomas que fornecem um sistema de derivação satisfatório (correto e completo) para a lógica de primeira ordem. Alguns são organizados de acordo com os conectivos que governam, por exemplo, os esquemas

$$A \to (B \to A) \qquad B \to (B \vee C) \qquad (B \wedge C) \to B$$

são axiomas comuns que governam $\to$, $\vee$ e $\wedge$. Alguns sistemas de axiomas visam um número mínimo de axiomas. Dependendo dos conectivos tomados como primitivos, é até possível encontrar sistemas de axiomas que consistem em um único axioma.

Uma regra de inferência é uma afirmação condicional que dá uma condição suficiente para que uma sentença em uma derivação seja justificada. O modus ponens é uma regra muito comum desse tipo: diz que se A e $A \to B$ já estão justificados, então B está justificado. Isso significa que uma linha em uma derivação que contém a sentença B está justificada, desde que tanto A quanto $A \to B$ (para alguma sentença A) apareçam na derivação antes de B.

A relação $\vdash$ baseada em derivações axiomáticas é definida da seguinte forma: $\Gamma \vdash A$ se, e somente se, há uma derivação com a sentença A como sua última fórmula (e Γ é tomado como o conjunto de sentenças nessa derivação que são justificadas por (2) acima). A é um teorema se A tem uma derivação em que Γ é vazio, isto é, toda sentença na derivação é justificada por (1) ou (3). Por exemplo, aqui está uma derivação que mostra que $\vdash A \to (B \to (B \vee A))$:

1. $B \to (B \vee A)$
2. $(B \to (B \vee A)) \to (A \to (B \to (B \vee A)))$
3. $A \to (B \to (B \vee A))$

A sentença na linha 1 tem a forma do axioma $A \to (A \vee B)$ (com os papéis de A e B invertidos). A sentença na linha 2 tem a forma do axioma $A \to (B \to A)$. Assim, ambas as linhas estão justificadas. A linha 3 é justificada por modus ponens: se a abreviarmos como D, então a linha 2 tem a forma $C \to D$, onde C é $B \to (B \vee A)$, isto é, a linha 1.

Um conjunto Γ é inconsistente se $\Gamma \vdash \bot$. Um sistema de axiomas completo também provará que $\bot \to A$ para qualquer A, e assim, se Γ é inconsistente, então $\Gamma \vdash A$ para qualquer A.

Sistemas de derivações axiomáticas para lógica foram dados pela primeira vez por Gottlob Frege em sua *Begriffsschrift* de 1879, que por essa razão é frequentemente considerada a primeira obra da lógica moderna. Foram aperfeiçoados em *Principia Mathematica* de Alfred North Whitehead e Bertrand Russell e por David Hilbert e seus alunos na década de 1920. São assim frequentemente chamados de "sistemas de Frege" ou "sistemas de Hilbert". São muito versáteis no sentido de que é frequentemente fácil encontrar um sistema axiomático para uma lógica. Porque as derivações têm uma estrutura muito simples e apenas uma ou duas regras de inferência, também é relativamente fácil provar coisas *sobre* eles. No entanto, são muito difíceis de usar na prática, isto é, é difícil encontrar e escrever provas.

CAPÍTULO 10

O Cálculo de Sequentes

10.1 Regras e Derivações

Para o que segue, sejam $\Gamma, \Delta, \Pi, \Lambda$ representando sequências finitas de sentenças.

Definição 10.1 (Sequente). Um *sequente* é uma expressão da forma

$$\Gamma \Rightarrow \Delta$$

em que Γ e Δ são sequências finitas (possivelmente vazias) de sentenças da linguagem $\mathscr{L}$. Γ é chamado de *antecedente*, enquanto Δ é o *sucedente*.

A ideia intuitiva por trás de um sequente é: se todas as sentenças no antecedente valem, então ao menos uma das sentenças no sucedente vale. Isto é, se $\Gamma = \langle A_1, \ldots, A_m \rangle$ e $\Delta = \langle B_1, \ldots, B_n \rangle$, então $\Gamma \Rightarrow \Delta$ vale se, e somente se,

$$(A_1 \wedge \cdots \wedge A_m) \rightarrow (B_1 \vee \cdots \vee B_n)$$

vale. Há dois casos especiais: quando Γ é vazio e quando Δ é vazio. Quando Γ é vazio, isto é, $m = 0$, $\Rightarrow \Delta$ vale se, e somente

se, $B_1 \vee \cdots \vee B_n$ vale. Quando Δ é vazio, isto é, $n = 0$, $\Gamma \Rightarrow$ vale se, e somente se, $\neg(A_1 \wedge \cdots \wedge A_m)$ vale. Dizemos que um sequente é válido se, e somente se, a sentença correspondente é válida.

Se Γ é uma sequência de sentenças, escrevemos Γ, A para o resultado de acrescentar A à extremidade direita de Γ (e A, Γ para o resultado de acrescentar A à extremidade esquerda de Γ). Se Δ também é uma sequência de sentenças, então Γ, Δ é a concatenação das duas sequências.

Definição 10.2 (Sequente Inicial). Um *sequente inicial* é um sequente de uma das seguintes formas:

1. $A \Rightarrow A$
2. $\bot \Rightarrow$

para qualquer sentença A na linguagem.

Derivações no cálculo de sequentes são certas árvores de sequentes, em que os sequentes no topo são sequentes iniciais e, se um sequente está abaixo de um ou dois outros sequentes, ele deve seguir corretamente por uma regra de inferência. As regras para **LK** dividem-se em dois tipos principais: regras *lógicas* e regras *estruturais*. As regras lógicas recebem o nome do operador principal da sentença que contém A e/ou B no sequente inferior. Cada uma vem em duas versões, uma para inferir um sequente com a sentença que contém o operador à esquerda, e uma com a sentença à direita.

10.2 Regras Proposicionais

Regras para ¬

$$\frac{\Gamma \Rightarrow \Delta, A}{\neg A, \Gamma \Rightarrow \Delta}\ \neg\text{L} \qquad \frac{A, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg A}\ \neg\text{R}$$

Regras para $\wedge$

$$\frac{A, \Gamma \Rightarrow \Delta}{A \wedge B, \Gamma \Rightarrow \Delta} \wedge\text{L} \qquad \frac{B, \Gamma \Rightarrow \Delta}{A \wedge B, \Gamma \Rightarrow \Delta} \wedge\text{L} \qquad \frac{\Gamma \Rightarrow \Delta, A \quad \Gamma \Rightarrow \Delta, B}{\Gamma \Rightarrow \Delta, A \wedge B} \wedge\text{R}$$

Regras para $\vee$

$$\frac{A, \Gamma \Rightarrow \Delta \quad B, \Gamma \Rightarrow \Delta}{A \vee B, \Gamma \Rightarrow \Delta} \vee\text{L} \qquad \frac{\Gamma \Rightarrow \Delta, A}{\Gamma \Rightarrow \Delta, A \vee B} \vee\text{R} \qquad \frac{\Gamma \Rightarrow \Delta, B}{\Gamma \Rightarrow \Delta, A \vee B} \vee\text{R}$$

Regras para $\rightarrow$

$$\frac{\Gamma \Rightarrow \Delta, A \quad B, \Pi \Rightarrow \Lambda}{A \rightarrow B, \Gamma, \Pi \Rightarrow \Delta, \Lambda} \rightarrow\text{L} \qquad \frac{A, \Gamma \Rightarrow \Delta, B}{\Gamma \Rightarrow \Delta, A \rightarrow B} \rightarrow\text{R}$$

10.3 Regras de Quantificadores

Regras para $\forall$

$$\frac{A(t), \Gamma \Rightarrow \Delta}{\forall x\, A(x), \Gamma \Rightarrow \Delta} \forall\text{L} \qquad \frac{\Gamma \Rightarrow \Delta, A(a)}{\Gamma \Rightarrow \Delta, \forall x\, A(x)} \forall\text{R}$$

Em $\forall$L, t é um termo fechado (isto é, um termo sem variáveis). Em $\forall$R, a é um símbolo de constante que não pode ocorrer em lugar algum do sequente inferior da regra $\forall$R. Chamamos a de *variável própria* da inferência $\forall$R.[1]

[1] Usamos o termo "variável própria" embora a na regra acima seja um símbolo de constante. Isso tem razões históricas.

Regras para ∃

$$\frac{A(a), \Gamma \Rightarrow \Delta}{\exists x\, A(x), \Gamma \Rightarrow \Delta}\ \exists\mathrm{L} \qquad\qquad \frac{\Gamma \Rightarrow \Delta, A(t)}{\Gamma \Rightarrow \Delta, \exists x\, A(x)}\ \exists\mathrm{R}$$

Novamente, t é um termo fechado, e a é um símbolo de constante que não ocorre no sequente inferior da regra ∃L. Chamamos a de *variável própria* da inferência ∃L.

A condição de que uma variável própria não ocorra no sequente inferior da inferência ∀R ou ∃L é chamada de *condição de variável própria.*

Lembre-se da convenção de que, quando A é uma fórmula com a variável x livre, indicamos isso escrevendo $A(x)$. No mesmo contexto, $A(t)$ é então uma abreviação de $A[t/x]$. Assim, poderíamos também escrever a regra ∃R como:

$$\frac{\Gamma \Rightarrow \Delta, A[t/x]}{\Gamma \Rightarrow \Delta, \exists x\, A}\ \exists\mathrm{R}$$

Note que t pode já ocorrer em A, por exemplo, A poderia ser $P(t,x)$. Assim, inferir $\Gamma \Rightarrow \Delta, \exists x\, P(t,x)$ de $\Gamma \Rightarrow \Delta, P(t,t)$ é uma aplicação correta de ∃R—pode-se "substituir" uma ou mais, e não necessariamente todas, as ocorrências de t na premissa pela variável x ligada. Contudo, as condições de variável própria em ∀R e ∃L exigem que a símbolo de constante a não ocorra em A. Portanto, não se pode inferir corretamente $\Gamma \Rightarrow \Delta, \forall x\, P(a,x)$ de $\Gamma \Rightarrow \Delta, P(a,a)$ usando ∀R.

Em ∃R e ∀L não há restrições sobre o termo t. Por outro lado, nas regras ∃L e ∀R, a condição de variável própria exige que a símbolo de constante a não ocorra em lugar algum fora de $A(a)$ no sequente superior. Isso é necessário para garantir que o sistema seja correto, isto é, que só deriva sequentes que sejam válidos. Sem essa condição, o seguinte seria permitido:

$$\dfrac{\dfrac{A(a) \Rightarrow A(a)}{\exists x\, A(x) \Rightarrow A(a)}\ {*}\exists\mathrm{L}}{\exists x\, A(x) \Rightarrow \forall x\, A(x)}\ \forall\mathrm{R} \qquad\qquad \dfrac{\dfrac{A(a) \Rightarrow A(a)}{A(a) \Rightarrow \forall x\, A(x)}\ {*}\forall\mathrm{R}}{\exists x\, A(x) \Rightarrow \forall x\, A(x)}\ \exists\mathrm{L}$$

Contudo, $\exists x\, A(x) \Rightarrow \forall x\, A(x)$ não é válido.

10.4 Regras Estruturais

Precisamos também de algumas regras que nos permitam reorganizar sentenças nos lados esquerdo e direito de um sequente. Como as regras lógicas exigem que as sentenças na premissa sobre as quais a regra atua estejam ou na extremidade esquerda ou na extremidade direita, precisamos de uma regra de "troca" que nos permita mover sentenças para a posição correta. Às vezes também é importante poder combinar duas sentenças idênticas em uma só, e acrescentar uma sentença em qualquer um dos lados.

Enfraquecimento

$$\frac{\Gamma \Rightarrow \Delta}{A, \Gamma \Rightarrow \Delta}\ \text{WL} \qquad \frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, A}\ \text{WR}$$

Contração

$$\frac{A, A, \Gamma \Rightarrow \Delta}{A, \Gamma \Rightarrow \Delta}\ \text{CL} \qquad \frac{\Gamma \Rightarrow \Delta, A, A}{\Gamma \Rightarrow \Delta, A}\ \text{CR}$$

Troca

$$\frac{\Gamma, A, B, \Pi \Rightarrow \Delta}{\Gamma, B, A, \Pi \Rightarrow \Delta}\ \text{XL} \qquad \frac{\Gamma \Rightarrow \Delta, A, B, \Lambda}{\Gamma \Rightarrow \Delta, B, A, \Lambda}\ \text{XR}$$

Uma série de inferências de enfraquecimento, contração e troca será frequentemente indicada por linhas de inferência duplas.

A regra a seguir, chamada de "corte", não é estritamente necessária, mas torna muito mais fácil reutilizar e combinar derivações.

$$\frac{\Gamma \Rightarrow \Delta, A \qquad A, \Pi \Rightarrow \Lambda}{\Gamma, \Pi \Rightarrow \Delta, \Lambda}\ \text{Corte}$$

10.5 Derivações

Já dissemos como é um sequente inicial e demos as regras de inferência. Derivações no cálculo de sequentes são geradas indutivamente a partir disso: cada derivação ou é um sequente inicial por si só, ou consiste em uma ou duas derivações seguidas de uma inferência.

Definição 10.3 (LK-derivação). Uma **LK**-*derivação* de um sequente S é uma árvore finita de sequentes que satisfaz as seguintes condições:

1. Os sequentes no topo da árvore são sequentes iniciais.
2. O sequente na base da árvore é S.
3. Todo sequente na árvore exceto S é uma premissa de uma aplicação correta de uma regra de inferência cuja conclusão está diretamente abaixo daquele sequente na árvore.

Dizemos então que S é o *sequente final* da derivação e que S é *derivável em* **LK** (ou **LK**-derivável).

Exemplo 10.4. Todo sequente inicial, por exemplo, $C \Rightarrow C$, é uma derivação. Podemos obter uma nova derivação a partir disso aplicando, digamos, a regra WL,

$$\frac{\Gamma \Rightarrow \Delta}{A, \Gamma \Rightarrow \Delta}\ \text{WL}$$

A regra, contudo, deve ser geral: podemos substituir o A na regra por qualquer sentença, por exemplo, também por D. Se a

premissa corresponde ao nosso sequente inicial $C \Rightarrow C$, isso significa que tanto Γ quanto Δ são apenas C, e a conclusão seria então $D, C \Rightarrow C$. Assim, o seguinte é uma derivação:

$$\dfrac{C \Rightarrow C}{D, C \Rightarrow C}\,\text{WL}$$

Podemos agora aplicar outra regra, digamos XL, que nos permite trocar duas sentenças à esquerda. Assim, o seguinte também é uma derivação correta:

$$\dfrac{\dfrac{C \Rightarrow C}{D, C \Rightarrow C}\,\text{WL}}{C, D \Rightarrow C}\,\text{XL}$$

Nesta aplicação da regra, que foi dada como

$$\dfrac{\Gamma, A, B, \Pi \Rightarrow \Delta}{\Gamma, B, A, \Pi \Rightarrow \Delta,}\,\text{XL}$$

tanto Γ quanto Π eram vazios, Δ é C, e os papéis de A e B são desempenhados por D e C, respectivamente. De modo muito semelhante, vemos também que

$$\dfrac{D \Rightarrow D}{C, D \Rightarrow D}\,\text{WL}$$

é uma derivação. Agora podemos pegar essas duas derivações e combiná-las usando $\wedge$R. Aquela regra era

$$\dfrac{\Gamma \Rightarrow \Delta, A \qquad \Gamma \Rightarrow \Delta, B}{\Gamma \Rightarrow \Delta, A \wedge B}\,\wedge\text{R}$$

No nosso caso, as premissas devem corresponder aos últimos sequentes das derivações que terminam nas premissas. Isso significa que Γ é C, D, Δ é vazio, A é C e B é D. Assim, a conclusão, se a inferência for correta, é $C, D \Rightarrow C \wedge D$.

$$\dfrac{\dfrac{\dfrac{C \Rightarrow C}{D, C \Rightarrow C}\,\text{WL}}{C, D \Rightarrow C}\,\text{XL} \qquad \dfrac{D \Rightarrow D}{C, D \Rightarrow D}\,\text{WL}}{C, D \Rightarrow C \wedge D}\,\wedge\text{R}$$

É claro que também podemos inverter as premissas; então A seria D e B seria C.

$$\dfrac{\dfrac{\dfrac{D \Rightarrow D}{C, D \Rightarrow D}\,\mathrm{WL} \qquad \dfrac{\dfrac{C \Rightarrow C}{D, C \Rightarrow C}\,\mathrm{WL}}{C, D \Rightarrow C}\,\mathrm{XL}}{C, D \Rightarrow D \wedge C}\,\wedge\mathrm{R}}$$

10.6 Exemplos de Derivações

Exemplo 10.5. Dê uma **LK**-derivação para o sequente $A \wedge B \Rightarrow A$.

Começamos escrevendo o sequente final desejado na base da derivação.

$$\dfrac{}{A \wedge B \Rightarrow A}$$

Em seguida, precisamos descobrir que tipo de inferência poderia ter um sequente inferior dessa forma. Poderia ser uma regra estrutural, mas é uma boa ideia começar procurando uma regra lógica. O único conectivo lógico que ocorre no sequente inferior é $\wedge$, então procuramos uma regra de $\wedge$ e, como o símbolo $\wedge$ ocorre no antecedente, estamos diante da regra $\wedge$L.

$$\dfrac{}{A \wedge B \Rightarrow A}\,\wedge\mathrm{L}$$

Há duas opções para o que poderia ter sido o sequente superior da inferência $\wedge$L: poderíamos ter um sequente superior $A \Rightarrow A$ ou $B \Rightarrow A$. Claramente, $A \Rightarrow A$ é um sequente inicial (o que é bom), enquanto $B \Rightarrow A$ não é derivável em geral. Preenchemos o sequente superior:

$$\dfrac{A \Rightarrow A}{A \wedge B \Rightarrow A}\,\wedge\mathrm{L}$$

Agora temos uma **LK**-derivação correta do sequente $A \wedge B \Rightarrow A$.

Exemplo 10.6. Dê uma **LK**-derivação para o sequente $\neg A \vee B \Rightarrow A \to B$.

Comece escrevendo o sequente final desejado na base da derivação.

$$\dfrac{}{\neg A \vee B \Rightarrow A \to B}$$

Para encontrar uma regra lógica que pudesse nos dar esse sequente final, olhamos para os conectivos lógicos no sequente final: $\neg$, $\vee$ e $\to$. No momento, só nos importam $\vee$ e $\to$, porque são operadores principais de sentenças no sequente final, enquanto $\neg$ está dentro do escopo de outro conectivo, de modo que cuidaremos dele depois. Nossas opções de regras lógicas para a inferência final são, portanto, a regra $\vee$L e a regra $\to$R. Poderíamos escolher qualquer uma das regras, na verdade, mas vamos escolher a regra $\to$R (quando não fosse por outro motivo, porque ela nos permite adiar a divisão em dois ramos). De acordo com a forma das inferências $\to$R que podem produzir o sequente inferior, isto deve ser assim:

$$\dfrac{\dfrac{}{A, \neg A \vee B \Rightarrow B}}{\neg A \vee B \Rightarrow A \to B} \to\text{R}$$

Se movermos $\neg A \vee B$ para a parte externa do antecedente, podemos aplicar a regra $\vee$L. De acordo com o esquema, isto deve se dividir em dois sequentes superiores como segue:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\neg A, A \Rightarrow B} \quad \dfrac{}{B, A \Rightarrow B}}{\neg A \vee B, A \Rightarrow B} \vee\text{L}}{A, \neg A \vee B \Rightarrow B}\,\text{XR}}{\neg A \vee B \Rightarrow A \to B} \to\text{R}$$

Lembre-se de que estamos tentando subir até sequentes iniciais; parece que estamos bem perto! O ramo direito está a apenas um enfraquecimento e uma troca de um sequente inicial, e então estará pronto:

$$\dfrac{\dfrac{\dfrac{}{\neg A, A \Rightarrow B} \quad \dfrac{\dfrac{\dfrac{B \Rightarrow B}{A, B \Rightarrow B}\,\mathrm{WL}}{B, A \Rightarrow B}\,\mathrm{XL}}{\neg A \vee B, A \Rightarrow B}\,\vee\mathrm{L}}{A, \neg A \vee B \Rightarrow B}\,\mathrm{XR}}{\neg A \vee B \Rightarrow A \to B}\,{\to}\mathrm{R}$$

Agora, olhando para o ramo esquerdo, o único conectivo lógico em qualquer sentença é o símbolo $\neg$ nas sentenças do antecedente, então estamos diante de uma instância da regra $\neg$L.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A \Rightarrow B, A}}{\neg A, A \Rightarrow B}\,\neg\mathrm{L} \quad \dfrac{\dfrac{\dfrac{B \Rightarrow B}{A, B \Rightarrow B}\,\mathrm{WL}}{B, A \Rightarrow B}\,\mathrm{XL}}{\neg A \vee B, A \Rightarrow B}\,\vee\mathrm{L}}{A, \neg A \vee B \Rightarrow B}\,\mathrm{XR}}{\neg A \vee B \Rightarrow A \to B}\,{\to}\mathrm{R}$$

De modo semelhante a como finalizamos o ramo direito, estamos a apenas um enfraquecimento e uma troca de finalizar também este ramo esquerdo.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \Rightarrow A}{A \Rightarrow A, B}\,\mathrm{WR}}{A \Rightarrow B, A}\,\mathrm{XR}}{\neg A, A \Rightarrow B}\,\neg\mathrm{L} \quad \dfrac{\dfrac{\dfrac{B \Rightarrow B}{A, B \Rightarrow B}\,\mathrm{WL}}{B, A \Rightarrow B}\,\mathrm{XL}}{\neg A \vee B, A \Rightarrow B}\,\vee\mathrm{L}}{A, \neg A \vee B \Rightarrow B}\,\mathrm{XR}}{\neg A \vee B \Rightarrow A \to B}\,{\to}\mathrm{R}$$

Exemplo 10.7. Dê uma **LK**-derivação do sequente $\neg A \vee \neg B \Rightarrow \neg(A \wedge B)$

Usando as técnicas acima, começamos escrevendo o sequente final desejado na base.

$$\dfrac{}{\neg A \vee \neg B \Rightarrow \neg(A \wedge B)}$$

Os conectivos principais disponíveis das sentenças no sequente final são o símbolo $\vee$ e o símbolo $\neg$. Funcionaria aplicar aqui

ou a regra ∨L ou a regra ¬R, mas começamos com a regra ¬R porque ela evita, por ora, a divisão em dois ramos:

$$\dfrac{\dfrac{}{A \wedge B, \neg A \vee \neg B \Rightarrow}}{\neg A \vee \neg B \Rightarrow \neg(A \wedge B)}\,\neg\mathrm{R}$$

Agora temos a escolha de olhar para a regra ∧L ou a regra ∨L. Vejamos o que acontece quando aplicamos a regra ∧L: temos a escolha de começar com o sequente $A, \neg A \vee B \Rightarrow$ ou com o sequente $B, \neg A \vee B \Rightarrow$. Como a derivação é simétrica em relação a A e B, vamos com o primeiro:

$$\dfrac{\dfrac{\dfrac{}{A, \neg A \vee \neg B \Rightarrow}}{A \wedge B, \neg A \vee \neg B \Rightarrow}\,\wedge\mathrm{L}}{\neg A \vee \neg B \Rightarrow \neg(A \wedge B)}\,\neg\mathrm{R}$$

Continuando a preencher a derivação, vemos que esbarramos em um problema:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \Rightarrow A}{\neg A, A \Rightarrow}\,\neg\mathrm{L} \qquad \dfrac{\dfrac{}{A \Rightarrow B}\,?}{\neg B, A \Rightarrow}\,\neg\mathrm{L}}{\neg A \vee \neg B, A \Rightarrow}\,\vee\mathrm{L}}{A, \neg A \vee \neg B \Rightarrow}\,\mathrm{XL}}{A \wedge B, \neg A \vee \neg B \Rightarrow}\,\wedge\mathrm{L}}{\neg A \vee \neg B \Rightarrow \neg(A \wedge B)}\,\neg\mathrm{R}$$

O topo do ramo direito não pode ser reduzido mais, e ele não pode ser levado, por meio de inferências estruturais, a um sequente inicial, então este não é o caminho certo a tomar. Assim, claramente, foi um erro aplicar a regra ∧L acima. Voltando ao que tínhamos antes e executando a regra ∨L em vez disso, obtemos

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\neg A, A \wedge B \Rightarrow} \qquad \dfrac{}{\neg B, A \wedge B \Rightarrow}}{\neg A \vee \neg B, A \wedge B \Rightarrow}\,\vee\mathrm{L}}{A \wedge B, \neg A \vee \neg B \Rightarrow}\,\mathrm{XL}}{\neg A \vee \neg B \Rightarrow \neg(A \wedge B)}\,\neg\mathrm{R}$$

Completando cada ramo como fizemos antes, obtemos

$$\dfrac{\dfrac{\dfrac{\dfrac{A \Rightarrow A}{A \wedge B \Rightarrow A}\wedge\mathrm{L}}{\neg A, A \wedge B \Rightarrow}\neg\mathrm{L} \qquad \dfrac{\dfrac{B \Rightarrow B}{A \wedge B \Rightarrow B}\wedge\mathrm{L}}{\neg B, A \wedge B \Rightarrow}\neg\mathrm{L}}{\neg A \vee \neg B, A \wedge B \Rightarrow}\vee\mathrm{L}}{\dfrac{A \wedge B, \neg A \vee \neg B \Rightarrow}{\neg A \vee \neg B \Rightarrow \neg(A \wedge B)}\neg\mathrm{R}}\mathrm{XL}$$

(Poderíamos ter executado as regras de $\wedge$ mais abaixo do que as regras de $\neg$ nesses passos e ainda assim obtido uma derivação correta).

Exemplo 10.8. Até agora não usamos a regra de contração, mas às vezes ela é necessária. Eis um exemplo em que isso acontece. Suponha que queiramos provar $\Rightarrow A \vee \neg A$. Aplicar $\vee$R de trás para frente nos daria uma destas duas derivações:

$$\dfrac{\dfrac{}{\Rightarrow A}}{\Rightarrow A \vee \neg A}\vee\mathrm{R} \qquad \dfrac{\dfrac{\dfrac{}{A \Rightarrow}}{\Rightarrow \neg A}\neg\mathrm{R}}{\Rightarrow A \vee \neg A}\vee\mathrm{R}$$

Nenhuma destas, é claro, termina em um sequente inicial. O truque é perceber que a regra de contração nos permite combinar duas cópias de uma sentença em uma só—e quando estamos buscando uma prova, isto é, indo de baixo para cima, podemos manter uma cópia de $A \vee \neg A$ na premissa, por exemplo,

$$\dfrac{\dfrac{\dfrac{}{\Rightarrow A \vee \neg A, A}}{\Rightarrow A \vee \neg A, A \vee \neg A}\vee\mathrm{R}}{\Rightarrow A \vee \neg A}\mathrm{CR}$$

Agora podemos aplicar $\vee$R uma segunda vez, e também obter $\neg A$, o que leva a uma derivação completa.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \Rightarrow A}{\Rightarrow A, \neg A}\,\neg\mathrm{R}}{\Rightarrow A, A \vee \neg A}\,\vee\mathrm{R}}{\Rightarrow A \vee \neg A, A}\,\mathrm{XR}}{\Rightarrow A \vee \neg A, A \vee \neg A}\,\vee\mathrm{R}}{\Rightarrow A \vee \neg A}\,\mathrm{CR}$$

10.7 Derivações com Quantificadores

Exemplo 10.9. Dê uma **LK**-derivação do sequente $\exists x\, \neg A(x) \Rightarrow \neg \forall x\, A(x)$.

Ao lidar com quantificadores, temos de cuidar para não violar a condição de variável própria, e às vezes isso exige que brinquemos com a ordem de execução de certas inferências. Em geral, ajuda tentar cuidar primeiro das regras sujeitas à condição de variável própria (elas ficarão mais abaixo na prova finalizada). Além disso, é uma boa ideia tentar olhar adiante e adivinhar como o sequente inicial poderia ser. No nosso caso, ele terá de ser algo como $A(a) \Rightarrow A(a)$. Isso significa que, ao "reverter" as regras de quantificador, teremos de escolher o mesmo termo—que chamaremos de a—tanto para a regra $\forall$ quanto para a regra $\exists$. Se escolhêssemos termos diferentes para cada regra, acabaríamos com algo como $A(a) \Rightarrow A(b)$, que, é claro, não é derivável.

Começando como de costume, escrevemos

$$\dfrac{}{\exists x\, \neg A(x) \;\Rightarrow\; \neg \forall x\, A(x)}$$

Poderíamos executar a regra $\exists$L ou a regra $\neg$R. Como a regra $\exists$L está sujeita à condição de variável própria, é uma boa ideia cuidar dela mais cedo do que mais tarde, então faremos essa primeiro.

$$\dfrac{\dfrac{}{\neg A(a) \;\Rightarrow\; \neg \forall x\, A(x)}}{\exists x\, \neg A(x) \;\Rightarrow\; \neg \forall x\, A(x)}\,\exists\mathrm{L}$$

Aplicando as regras $\neg$L e $\neg$R de trás para frente, obtemos

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\forall x\, A(x) \Rightarrow A(a)}}{\neg A(a), \forall x\, A(x) \Rightarrow}\,\neg\mathrm{L}}{\forall x\, A(x), \neg A(a) \Rightarrow}\,\mathrm{XL}}{\neg A(a) \Rightarrow \neg\forall x A(x)}\,\neg\mathrm{R}}{\exists x \neg A(x) \Rightarrow \neg\forall x A(x)}\,\exists\mathrm{L}$$

Neste ponto, nossa única opção é executar a regra ∀L. Como essa regra não está sujeita à restrição de variável própria, estamos livres. Lembre-se, queremos tentar obter um sequente inicial (da forma $A(a) \Rightarrow A(a)$), então devemos escolher a como nosso argumento para A ao aplicarmos a regra.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A(a) \Rightarrow A(a)}}{\forall x\, A(x) \Rightarrow A(a)}\,\forall\mathrm{L}}{\neg A(a), \forall x\, A(x) \Rightarrow}\,\neg\mathrm{L}}{\forall x\, A(x), \neg A(a) \Rightarrow}\,\mathrm{XL}}{\neg A(a) \Rightarrow \neg\forall x\, A(x)}\,\neg\mathrm{R}}{\exists x\, \neg A(x) \Rightarrow \neg\forall x\, A(x)}\,\exists\mathrm{L}$$

É importante, especialmente ao lidar com quantificadores, verificar com cuidado neste ponto que a condição de variável própria não foi violada. Como a única regra que aplicamos sujeita à condição de variável própria foi ∃L, e a variável própria a não ocorre em seu sequente inferior (o sequente final), esta é uma derivação correta.

10.8 Noções da Teoria da Prova

Assim como definimos uma série de noções semânticas importantes (validade, consequência, satisfatibilidade), definimos agora as *noções da teoria da prova* correspondentes. Elas não são definidas por apelo à satisfação de sentenças em estruturas, mas por apelo à derivabilidade ou à não derivabilidade de certos sequentes. Foi uma descoberta importante que essas noções coincidem. Que elas coincidem é o conteúdo dos teoremas da *correção* e da *completude*.

Definição 10.10 (Teoremas). Uma sentença A é um *teorema* se há uma derivação em **LK** do sequente $\Rightarrow A$. Escrevemos $\vdash A$ se A é um teorema e $\nvdash A$ se não é.

Definição 10.11 (Derivabilidade). Uma sentença A é *derivável de* um conjunto de sentenças Γ, $\Gamma \vdash A$, se, e somente se, há um subconjunto finito $\Gamma_0 \subseteq \Gamma$ e uma sequência Γ_0' das sentenças em Γ_0 tal que **LK** deriva $\Gamma_0' \Rightarrow A$. Se A não é derivável de Γ, escrevemos $\Gamma \nvdash A$.

Por causa das regras de contração, enfraquecimento e troca, a ordem e o número de sentenças em Γ_0' não importam: se um sequente $\Gamma_0' \Rightarrow A$ é derivável, então também o é $\Gamma_0'' \Rightarrow A$ para qualquer Γ_0'' que contenha as mesmas sentenças que Γ_0'. Por exemplo, se $\Gamma_0 = \{B, C\}$, então tanto $\Gamma_0' = \langle B, B, C\rangle$ quanto $\Gamma_0'' = \langle C, C, B\rangle$ são sequências que contêm apenas as sentenças em Γ_0. Se um sequente contendo uma delas é derivável, o outro também é, por exemplo:

$$\dfrac{\dfrac{\dfrac{\vdots}{B, B, C \Rightarrow A}\,\text{CL}}{B, C \Rightarrow A}\,\text{XL}}{C, B \Rightarrow A}\,\text{WL}$$

De agora em diante, diremos que, se Γ_0 é um conjunto finito de sentenças, então $\Gamma_0 \Rightarrow A$ é qualquer sequente em que o antecedente é uma sequência de sentenças em Γ_0, e incluiremos tacitamente contrações, trocas e enfraquecimentos se necessário.

Definição 10.12 (Consistência). Um conjunto de sentenças Γ é *inconsistente* se, e somente se, há um subconjunto finito $\Gamma_0 \subseteq \Gamma$ tal que **LK** deriva $\Gamma_0 \Rightarrow$. Se Γ não é inconsistente, isto é, se

para todo $\Gamma_0 \subseteq \Gamma$ finito, **LK** não deriva $\Gamma_0 \Rightarrow$, dizemos que é *consistente.*

Proposição 10.13 (Reflexividade). *Se $A \in \Gamma$, então $\Gamma \vdash A$.*

Prova. O sequente inicial $A \Rightarrow A$ é derivável, e $\{A\} \subseteq \Gamma$. □

Proposição 10.14 (Monotonicidade). *Se $\Gamma \subseteq \Delta$ e $\Gamma \vdash A$, então $\Delta \vdash A$.*

Prova. Suponha $\Gamma \vdash A$, isto é, há um $\Gamma_0 \subseteq \Gamma$ finito tal que $\Gamma_0 \Rightarrow A$ é derivável. Como $\Gamma \subseteq \Delta$, então Γ_0 também é um subconjunto finito de Δ. A derivação de $\Gamma_0 \Rightarrow A$ mostra, assim, também que $\Delta \vdash A$. □

Proposição 10.15 (Transitividade). *Se $\Gamma \vdash A$ e $\{A\} \cup \Delta \vdash B$, então $\Gamma \cup \Delta \vdash B$.*

Prova. Se $\Gamma \vdash A$, há um $\Gamma_0 \subseteq \Gamma$ finito e uma derivação π_0 de $\Gamma_0 \Rightarrow A$. Se $\{A\} \cup \Delta \vdash B$, então, para algum subconjunto finito $\Delta_0 \subseteq \Delta$, há uma derivação π_1 de $A, \Delta_0 \Rightarrow B$. Considere a seguinte derivação:

$$\dfrac{\begin{array}{cc}\vdots\,\pi_0 & \vdots\,\pi_1 \\ \Gamma_0 \Rightarrow A & A, \Delta_0 \Rightarrow B\end{array}}{\Gamma_0, \Delta_0 \Rightarrow B}\ \text{Corte}$$

Como $\Gamma_0 \cup \Delta_0 \subseteq \Gamma \cup \Delta$, isso mostra $\Gamma \cup \Delta \vdash B$. □

Note que isso significa, em particular, que se $\Gamma \vdash A$ e $A \vdash B$, então $\Gamma \vdash B$. Segue-se também que se $A_1, \ldots, A_n \vdash B$ e $\Gamma \vdash A_i$ para cada i, então $\Gamma \vdash B$.

Proposição 10.16. *Γ é inconsistente se, e somente se, $\Gamma \vdash A$ para toda sentença A.*

Prova. Exercício. □

Proposição 10.17 (Compacidade). *1. Se $\Gamma \vdash A$, então há um subconjunto finito $\Gamma_0 \subseteq \Gamma$ tal que $\Gamma_0 \vdash A$.*

2. Se todo subconjunto finito de Γ é consistente, então Γ é consistente.

Prova. 1. Se $\Gamma \vdash A$, então há um subconjunto finito $\Gamma_0 \subseteq \Gamma$ tal que o sequente $\Gamma_0 \Rightarrow A$ tem uma derivação. Consequentemente, $\Gamma_0 \vdash A$.

2. Se Γ é inconsistente, há um subconjunto finito $\Gamma_0 \subseteq \Gamma$ tal que **LK** deriva $\Gamma_0 \Rightarrow$. Mas então Γ_0 é um subconjunto finito de Γ que é inconsistente. □

10.9 Derivabilidade e Consistência

Estabeleceremos agora uma série de propriedades da relação de derivabilidade. Elas são interessantes por si mesmas, mas cada uma terá um papel na prova do teorema da completude.

Proposição 10.18. *Se $\Gamma \vdash A$ e $\Gamma \cup \{A\}$ é inconsistente, então Γ é inconsistente.*

Prova. Há Γ_0 e $\Gamma_1 \subseteq \Gamma$ finitos tais que **LK** deriva $\Gamma_0 \Rightarrow A$ e $A, \Gamma_1 \Rightarrow$. Seja π_0 a **LK**-derivação de $\Gamma_0 \Rightarrow A$ e π_1 a **LK**-derivação de $\Gamma_1, A \Rightarrow$. Podemos então derivar

$$\dfrac{\begin{array}{cc}\vdots\,\pi_0 & \vdots\,\pi_1 \\ \Gamma_0 \Rightarrow A & A, \Gamma_1 \Rightarrow\end{array}}{\Gamma_0, \Gamma_1 \Rightarrow}\ \text{Corte}$$

Como $\Gamma_0 \subseteq \Gamma$ e $\Gamma_1 \subseteq \Gamma$, $\Gamma_0 \cup \Gamma_1 \subseteq \Gamma$; portanto, Γ é inconsistente. □

Proposição 10.19. *$\Gamma \vdash A$ se, e somente se, $\Gamma \cup \{\neg A\}$ é inconsistente.*

Prova. Primeiro, suponha $\Gamma \vdash A$, isto é, há uma derivação π_0 de $\Gamma \Rightarrow A$. Acrescentando uma regra ¬L, obtemos uma derivação de $\neg A, \Gamma \Rightarrow$, isto é, $\Gamma \cup \{\neg A\}$ é inconsistente.

Se $\Gamma \cup \{\neg A\}$ é inconsistente, há uma derivação π_1 de $\neg A, \Gamma \Rightarrow$. O seguinte é uma derivação de $\Gamma \Rightarrow A$:

$$\dfrac{\dfrac{A \Rightarrow A}{\Rightarrow A, \neg A}\,\neg\mathrm{R} \qquad \begin{matrix}\vdots\\ \pi_1\\ \vdots\end{matrix} \quad \neg A, \Gamma \Rightarrow}{\Gamma \Rightarrow A}\,\mathrm{Corte}$$

□

Proposição 10.20. *Se $\Gamma \vdash A$ e $\neg A \in \Gamma$, então Γ é inconsistente.*

Prova. Suponha $\Gamma \vdash A$ e $\neg A \in \Gamma$. Então há uma derivação π de um sequente $\Gamma_0 \Rightarrow A$. O sequente $\neg A, \Gamma_0 \Rightarrow$ também é derivável:

$$\dfrac{\begin{matrix}\vdots\\ \pi\\ \vdots\end{matrix} \quad \Gamma_0 \Rightarrow A \qquad \dfrac{\dfrac{A \Rightarrow A}{\neg A, A \Rightarrow}\,\neg\mathrm{L}}{A, \neg A \Rightarrow}\,\mathrm{XL}}{\Gamma_0, \neg A \Rightarrow}\,\mathrm{Corte}$$

Como $\neg A \in \Gamma$ e $\Gamma_0 \subseteq \Gamma$, isso mostra que Γ é inconsistente. □

Proposição 10.21. *Se $\Gamma \cup \{A\}$ e $\Gamma \cup \{\neg A\}$ são ambos inconsistentes, então Γ é inconsistente.*

Prova. Há conjuntos finitos $\Gamma_0 \subseteq \Gamma$ e $\Gamma_1 \subseteq \Gamma$ e **LK**-derivações π_0 e π_1 de $A, \Gamma_0 \Rightarrow$ e $\neg A, \Gamma_1 \Rightarrow$, respectivamente. Podemos então derivar

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \pi_0\end{array}}{\dfrac{A, \Gamma_0 \Rightarrow}{\Gamma_0 \Rightarrow \neg A}\,\neg\mathrm{R}} \qquad \dfrac{\begin{array}{c}\vdots\\ \pi_1\end{array}}{\neg A, \Gamma_1 \Rightarrow}}{\Gamma_0, \Gamma_1 \Rightarrow}\,\text{Corte}$$

Como $\Gamma_0 \subseteq \Gamma$ e $\Gamma_1 \subseteq \Gamma$, $\Gamma_0 \cup \Gamma_1 \subseteq \Gamma$. Portanto, Γ é inconsistente.□

10.10 Derivabilidade e os Conectivos Proposicionais

Estabelecemos que a relação de derivabilidade $\vdash$ do cálculo de sequentes é forte o suficiente para estabelecer alguns fatos básicos envolvendo os conectivos proposicionais, tais como $A \wedge B \vdash A$ e $A, A \to B \vdash B$ (modus ponens). Esses fatos são necessários para a prova do teorema da completude.

Proposição 10.22. *1. Tanto $A \wedge B \vdash A$ quanto $A \wedge B \vdash B$.*

2. $A, B \vdash A \wedge B$.

Prova. 1. Ambos os sequentes $A \wedge B \Rightarrow A$ e $A \wedge B \Rightarrow B$ são deriváveis:

$$\dfrac{A \Rightarrow A}{A \wedge B \Rightarrow A}\,\wedge\mathrm{L} \qquad \dfrac{B \Rightarrow B}{A \wedge B \Rightarrow B}\,\wedge\mathrm{L}$$

2. Eis uma derivação do sequente $A, B \Rightarrow A \wedge B$:

$$\dfrac{A \Rightarrow A \qquad B \Rightarrow B}{A, B \Rightarrow A \wedge B}\,\wedge\mathrm{R}$$

□

Proposição 10.23. *1. $A \vee B, \neg A, \neg B$ é inconsistente.*

2. Tanto $A \vdash A \vee B$ quanto $B \vdash A \vee B$.

Prova. 1. Damos uma derivação do sequente $A \vee B, \neg A, \neg B \Rightarrow$:

$$\dfrac{\dfrac{\dfrac{A \Rightarrow A}{\neg A, A \Rightarrow}\,\neg\mathrm{L}}{A, \neg A, \neg B \Rightarrow} \qquad \dfrac{\dfrac{B \Rightarrow B}{\neg B, B \Rightarrow}\,\neg\mathrm{L}}{B, \neg A, \neg B \Rightarrow}}{A \vee B, \neg A, \neg B \Rightarrow}\,\vee\mathrm{L}$$

(Lembre-se de que linhas de inferência duplas indicam várias inferências de enfraquecimento, contração e troca.)

2. Ambos os sequentes $A \Rightarrow A \vee B$ e $B \Rightarrow A \vee B$ têm derivações:

$$\dfrac{A \Rightarrow A}{A \Rightarrow A \vee B}\,\vee\mathrm{R} \qquad \dfrac{B \Rightarrow B}{B \Rightarrow A \vee B}\,\vee\mathrm{R}$$

□

Proposição 10.24. *1.* $A, A \to B \vdash B$.

2. Tanto $\neg A \vdash A \to B$ *quanto* $B \vdash A \to B$.

Prova. 1. O sequente $A \to B, A \Rightarrow B$ é derivável:

$$\dfrac{A \Rightarrow A \qquad B \Rightarrow B}{A \to B, A \Rightarrow B}\,\to\mathrm{L}$$

2. Ambos os sequentes $\neg A \Rightarrow A \to B$ e $B \Rightarrow A \to B$ são deriváveis:

$$\dfrac{\dfrac{\dfrac{\dfrac{A \Rightarrow A}{\neg A, A \Rightarrow}\,\neg\mathrm{L}}{A, \neg A \Rightarrow}\,\mathrm{XL}}{A, \neg A \Rightarrow B}\,\mathrm{WR}}{\neg A \Rightarrow A \to B}\,\to\mathrm{R} \qquad \dfrac{\dfrac{\dfrac{B \Rightarrow B}{A, B \Rightarrow B}\,\mathrm{WL}}{B \Rightarrow A \to B}\,\to\mathrm{R}}{}$$

□

10.11 Derivabilidade e os Quantificadores

O teorema da completude também exige que as regras do cálculo de sequentes produzam os fatos sobre $\vdash$ estabelecidos nesta seção.

Teorema 10.25. *Se c é uma constante que não ocorre em Γ ou $A(x)$ e $\Gamma \vdash A(c)$, então $\Gamma \vdash \forall x\, A(x)$.*

Prova. Seja π_0 uma **LK**-derivação de $\Gamma_0 \Rightarrow A(c)$ para algum $\Gamma_0 \subseteq \Gamma$ finito. Acrescentando uma inferência $\forall$R, obtemos uma derivação de $\Gamma_0 \Rightarrow \forall x\, A(x)$, já que c não ocorre em Γ ou $A(x)$ e, assim, a condição de variável própria é satisfeita. □

Proposição 10.26. 1. $A(t) \vdash \exists x\, A(x)$.

2. $\forall x\, A(x) \vdash A(t)$.

Prova. 1. O sequente $A(t) \Rightarrow \exists x\, A(x)$ é derivável:

$$\dfrac{A(t) \Rightarrow A(t)}{A(t) \Rightarrow \exists x\, A(x)}\,\exists\text{R}$$

2. O sequente $\forall x\, A(x) \Rightarrow A(t)$ é derivável:

$$\dfrac{A(t) \Rightarrow A(t)}{\forall x\, A(x) \Rightarrow A(t)}\,\forall\text{L}$$

□

10.12 Correção

Um sistema de derivação, tal como o cálculo de sequentes, é *correto* se não pode derivar coisas que de fato não valem. A correção é, assim, uma espécie de propriedade de segurança garantida para sistemas de derivação. Dependendo de qual propriedade da teoria da prova está em questão, gostaríamos de saber, por exemplo, que

1. todo A derivável é válido;

2. se uma sentença é derivável de algumas outras, ela também é consequência delas;

3. se um conjunto de sentenças é inconsistente, ele é insatisfatível.

Essas são propriedades importantes de um sistema de derivação. Se alguma delas não vale, o sistema de derivação é deficiente—ele deriva demais. Consequentemente, estabelecer a correção de um sistema de derivação é da máxima importância.

Como todas essas propriedades da teoria da prova são definidas via derivabilidade no cálculo de sequentes de certos sequentes, provar (1)–(3) acima requer provar algo sobre as propriedades semânticas dos sequentes deriváveis. Primeiro definiremos o que significa um sequente ser *válido*, e então mostraremos que todo sequente derivável é válido. (1)–(3) seguem-se então como corolários desse resultado.

Definição 10.27. Uma estrutura $\mathfrak{M}$ *satisfaz* um sequente $\Gamma \Rightarrow \Delta$ se, e somente se, ou $\mathfrak{M} \nvDash A$ para algum $A \in \Gamma$ ou $\mathfrak{M} \vDash A$ para algum $A \in \Delta$.

Um sequente é *válido* se, e somente se, toda estrutura $\mathfrak{M}$ o satisfaz.

Teorema 10.28 (Correção). *Se* **LK** *deriva* $\Theta \Rightarrow \Xi$, *então* $\Theta \Rightarrow \Xi$ *é válido.*

Prova. Seja π uma derivação de $\Theta \Rightarrow \Xi$. Procedemos por indução sobre o número de inferências n em π.

Se o número de inferências é 0, então π consiste apenas em um sequente inicial. Todo sequente inicial $A \Rightarrow A$ é obviamente válido, pois para toda $\mathfrak{M}$, ou $\mathfrak{M} \nvDash A$ ou $\mathfrak{M} \vDash A$.

Se o número de inferências é maior que 0, distinguimos casos de acordo com o tipo da inferência mais inferior. Pela hipótese de indução, podemos supor que as premissas dessa inferência são válidas, já que o número de inferências na derivação de qualquer premissa é menor que n.

Primeiro, consideramos as inferências possíveis com apenas uma premissa.

1. A última inferência é um enfraquecimento. Então $\Theta \Rightarrow \Xi$ é $A, \Gamma \Rightarrow \Delta$ (se a última inferência é WL) ou $\Gamma \Rightarrow \Delta, A$ (se é WR), e a derivação termina em uma de

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta\end{array}}{A, \Gamma \Rightarrow \Delta}\ \text{WL} \qquad \dfrac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta\end{array}}{\Gamma \Rightarrow \Delta, A}\ \text{WR}$$

   Pela hipótese de indução, $\Gamma \Rightarrow \Delta$ é válido, isto é, para toda estrutura $\mathfrak{M}$, ou há algum $C \in \Gamma$ tal que $\mathfrak{M} \nvDash C$ ou há algum $C \in \Delta$ tal que $\mathfrak{M} \vDash C$.

   Se $\mathfrak{M} \nvDash C$ para algum $C \in \Gamma$, então $C \in \Theta$ também, já que $\Theta = A, \Gamma$, e assim $\mathfrak{M} \nvDash C$ para algum $C \in \Theta$. Analogamente, se $\mathfrak{M} \vDash C$ para algum $C \in \Delta$, como $C \in \Xi$, $\mathfrak{M} \vDash C$ para algum $C \in \Xi$. Consequentemente, $\Theta \Rightarrow \Xi$ é válido.

2. A última inferência é ¬L: Então a premissa da última inferência é $\Gamma \Rightarrow \Delta, A$ e a conclusão é $\neg A, \Gamma \Rightarrow \Delta$, isto é, a derivação termina em

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta, A\end{array}}{\neg A, \Gamma \Rightarrow \Delta}\ \neg\text{L}$$

   e $\Theta = \neg A, \Gamma$ enquanto $\Xi = \Delta$.

   A hipótese de indução nos diz que $\Gamma \Rightarrow \Delta, A$ é válido, isto é, para toda $\mathfrak{M}$, ou (a) para algum $C \in \Gamma$, $\mathfrak{M} \nvDash C$, ou (b) para algum $C \in \Delta$, $\mathfrak{M} \vDash C$, ou (c) $\mathfrak{M} \vDash A$. Queremos mostrar que $\Theta \Rightarrow \Xi$ também é válido. Seja $\mathfrak{M}$ uma estrutura. Se (a) vale, então há $C \in \Gamma$ tal que $\mathfrak{M} \nvDash C$, mas $C \in \Theta$ também. Se (b) vale, há $C \in \Delta$ tal que $\mathfrak{M} \vDash C$, mas $C \in \Xi$ também. Por fim, se $\mathfrak{M} \vDash A$, então $\mathfrak{M} \nvDash \neg A$. Como $\neg A \in \Theta$, há $C \in \Theta$ tal que $\mathfrak{M} \nvDash C$. Consequentemente, $\Theta \Rightarrow \Xi$ é válido.

3. A última inferência é ¬R: Exercício.

4. A última inferência é ∧L: Há duas variantes: $A \wedge B$ pode ser inferido à esquerda a partir de A ou de B no lado esquerdo da premissa. No primeiro caso, π termina em

$$\dfrac{\begin{array}{c}\vdots\\ A, \Gamma \Rightarrow \Delta\end{array}}{A \wedge B, \Gamma \Rightarrow \Delta}\ \wedge\mathrm{L}$$

e $\Theta = A \wedge B, \Gamma$ enquanto $\Xi = \Delta$. Considere uma estrutura M. Como, pela hipótese de indução, $A, \Gamma \Rightarrow \Delta$ é válido, (a) $M \nvDash A$, (b) $M \nvDash C$ para algum $C \in \Gamma$, ou (c) $M \vDash C$ para algum $C \in \Delta$. No caso (a), $M \nvDash A \wedge B$, então há $C \in \Theta$ (a saber, $A \wedge B$) tal que $M \nvDash C$. No caso (b), há $C \in \Gamma$ tal que $M \nvDash C$, e $C \in \Theta$ também. No caso (c), há $C \in \Delta$ tal que $M \vDash C$, e $C \in \Xi$ também, já que $\Xi = \Delta$. Assim, em cada caso, M satisfaz $A \wedge B, \Gamma \Rightarrow \Delta$. Como M era arbitrário, $\Gamma \Rightarrow \Delta$ é válido. O caso em que $A \wedge B$ é inferido a partir de B é tratado da mesma forma, mudando A para B.

5. A última inferência é ∨R: Há duas variantes: $A \vee B$ pode ser inferido à direita a partir de A ou de B no lado direito da premissa. No primeiro caso, π termina em

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta, A\end{array}}{\Gamma \Rightarrow \Delta, A \vee B}\ \vee\mathrm{R}$$

Agora $\Theta = \Gamma$ e $\Xi = \Delta, A \vee B$. Considere uma estrutura M. Como $\Gamma \Rightarrow \Delta, A$ é válido, (a) $M \vDash A$, (b) $M \nvDash C$ para algum $C \in \Gamma$, ou (c) $M \vDash C$ para algum $C \in \Delta$. No caso (a), $M \vDash A \vee B$. No caso (b), há $C \in \Gamma$ tal que $M \nvDash C$. No caso (c), há $C \in \Delta$ tal que $M \vDash C$. Assim, em cada

caso, M satisfaz $\Gamma \Rightarrow \Delta, A \vee B$, isto é, $\Theta \Rightarrow \Xi$. Como M era arbitrário, $\Theta \Rightarrow \Xi$ é válido. O caso em que $A \vee B$ é inferido a partir de B é tratado da mesma forma, mudando A para B.

6. A última inferência é $\rightarrow$R: Então π termina em

$$\dfrac{\begin{array}{c}\vdots\\ A, \Gamma \Rightarrow \Delta, B\end{array}}{\Gamma \Rightarrow \Delta, A \rightarrow B} \rightarrow \mathrm{R}$$

Novamente, a hipótese de indução diz que a premissa é válida; queremos mostrar que a conclusão também é válida. Seja M arbitrário. Como $A, \Gamma \Rightarrow \Delta, B$ é válido, ao menos um dos seguintes casos ocorre: (a) $M \nvDash A$, (b) $M \vDash B$, (c) $M \nvDash C$ para algum $C \in \Gamma$, ou (d) $M \vDash C$ para algum $C \in \Delta$. Nos casos (a) e (b), $M \vDash A \rightarrow B$ e assim há um $C \in \Delta, A \rightarrow B$ tal que $M \vDash C$. No caso (c), para algum $C \in \Gamma$, $M \nvDash C$. No caso (d), para algum $C \in \Delta$, $M \vDash C$. Em cada caso, M satisfaz $\Gamma \Rightarrow \Delta, A \rightarrow B$. Como M era arbitrário, $\Gamma \Rightarrow \Delta, A \rightarrow B$ é válido.

7. A última inferência é $\forall$L: Então há uma fórmula $A(x)$ e um termo fechado t tais que π termina em

$$\dfrac{\begin{array}{c}\vdots\\ A(t), \Gamma \Rightarrow \Delta\end{array}}{\forall x\, A(x), \Gamma \Rightarrow \Delta} \forall \mathrm{L}$$

Queremos mostrar que a conclusão $\forall x\, A(x), \Gamma \Rightarrow \Delta$ é válida. Considere uma estrutura M. Como a premissa $A(t), \Gamma \Rightarrow \Delta$ é válida, (a) $M \nvDash A(t)$, (b) $M \nvDash C$ para algum $C \in \Gamma$, ou (c) $M \vDash C$ para algum $C \in \Delta$. No caso (a), por Proposição 7.31, se $M \vDash \forall x\, A(x)$, então $M \vDash A(t)$.

Como $\mathfrak{M} \nvDash A(t)$, $\mathfrak{M} \nvDash \forall x\, A(x)$. Nos casos (b) e (c), $\mathfrak{M}$ também satisfaz $\forall x\, A(x), \Gamma \Rightarrow \Delta$. Como $\mathfrak{M}$ era arbitrário, $\forall x\, A(x), \Gamma \Rightarrow \Delta$ é válido.

8. A última inferência é $\exists$R: Exercício.

9. A última inferência é $\forall$R: Então há uma fórmula $A(x)$ e um símbolo de constante a tais que π termina em

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta, A(a)\end{array}}{\Gamma \Rightarrow \Delta, \forall x\, A(x)}\ \forall\text{R}$$

em que a condição de variável própria é satisfeita, isto é, a não ocorre em $A(x)$, Γ ou Δ. Pela hipótese de indução, a premissa da última inferência é válida. Temos de mostrar que a conclusão também é válida, isto é, que para qualquer estrutura $\mathfrak{M}$, (a) $\mathfrak{M} \vDash \forall x\, A(x)$, (b) $\mathfrak{M} \nvDash C$ para algum $C \in \Gamma$, ou (c) $\mathfrak{M} \vDash C$ para algum $C \in \Delta$.

Suponha que $\mathfrak{M}$ seja uma estrutura arbitrária. Se (b) ou (c) vale, terminamos, então suponha que nenhuma vale: para todo $C \in \Gamma$, $\mathfrak{M} \vDash C$, e para todo $C \in \Delta$, $\mathfrak{M} \nvDash C$. Temos de mostrar que (a) vale, isto é, $\mathfrak{M} \vDash \forall x\, A(x)$. Por Proposição 7.19, basta mostrar que $\mathfrak{M}, s \vDash A(x)$ para toda atribuição de variáveis s. Então seja s uma atribuição de variáveis arbitrária. Considere a estrutura $\mathfrak{M}'$ que é exatamente como $\mathfrak{M}$ exceto que $a^{\mathfrak{M}'} = s(x)$. Por Corolário 7.21, para qualquer $C \in \Gamma$, $\mathfrak{M}' \vDash C$ já que a não ocorre em Γ, e para qualquer $C \in \Delta$, $\mathfrak{M}' \nvDash C$. Mas a premissa é válida, então $\mathfrak{M}' \vDash A(a)$. Por Proposição 7.18, $\mathfrak{M}', s \vDash A(a)$, já que $A(a)$ é uma sentença. Agora $s \sim_x s$ com $s(x) = \mathrm{Val}_s^{\mathfrak{M}'}(a)$, já que definimos $\mathfrak{M}'$ exatamente desse modo. Assim Proposição 7.23 se aplica, e obtemos $\mathfrak{M}', s \vDash A(x)$. Como a não ocorre em $A(x)$, por Proposição 7.20, $\mathfrak{M}, s \vDash A(x)$. Como

s era arbitrário, completamos a prova de que $\boldsymbol{M}, s \vDash A(x)$ para toda atribuição de variáveis.

10. A última inferência é $\exists$L: Exercício.

Agora consideremos as inferências possíveis com duas premissas.

1. A última inferência é um corte: então π termina em

$$\dfrac{\begin{array}{cc}\vdots & \vdots \\ \Gamma \Rightarrow \Delta, A & A, \Pi \Rightarrow \Lambda\end{array}}{\Gamma, \Pi \Rightarrow \Delta, \Lambda}\ \text{Corte}$$

Seja $\boldsymbol{M}$ uma estrutura. Pela hipótese de indução, as premissas são válidas, então $\boldsymbol{M}$ satisfaz ambas as premissas. Distinguimos dois casos: (a) $\boldsymbol{M} \nvDash A$ e (b) $\boldsymbol{M} \vDash A$. No caso (a), para que $\boldsymbol{M}$ satisfaça a premissa esquerda, ela deve satisfazer $\Gamma \Rightarrow \Delta$. Mas então também satisfaz a conclusão. No caso (b), para que $\boldsymbol{M}$ satisfaça a premissa direita, ela deve satisfazer $\Pi \setminus \Lambda$. Novamente, $\boldsymbol{M}$ satisfaz a conclusão.

2. A última inferência é $\wedge$R. Então π termina em

$$\dfrac{\begin{array}{cc}\vdots & \vdots \\ \Gamma \Rightarrow \Delta, A & \Gamma \Rightarrow \Delta, B\end{array}}{\Gamma \Rightarrow \Delta, A \wedge B}\ \wedge\text{R}$$

Considere uma estrutura $\boldsymbol{M}$. Se $\boldsymbol{M}$ satisfaz $\Gamma \Rightarrow \Delta$, terminamos. Então suponha que não. Como $\Gamma \Rightarrow \Delta, A$ é válido pela hipótese de indução, $\boldsymbol{M} \vDash A$. Analogamente, como $\Gamma \Rightarrow \Delta, B$ é válido, $\boldsymbol{M} \vDash B$. Mas então $\boldsymbol{M} \vDash A \wedge B$.

3. A última inferência é $\vee$L: Exercício.

4. A última inferência é $\rightarrow$L. Então π termina em

$$\frac{\vdots \qquad\qquad \vdots}{\Gamma \Rightarrow \Delta, A \qquad B, \Pi \Rightarrow \Lambda}\ {\to}\text{L}$$
$$A \to B, \Gamma, \Pi \Rightarrow \Delta, \Lambda$$

Novamente, considere uma estrutura M e suponha que M não satisfaça $\Gamma, \Pi \Rightarrow \Delta, \Lambda$. Temos de mostrar que $M \nvDash A \to B$. Se M não satisfaz $\Gamma, \Pi \Rightarrow \Delta, \Lambda$, não satisfaz nem $\Gamma \Rightarrow \Delta$ nem $\Pi \Rightarrow \Lambda$. Como $\Gamma \Rightarrow \Delta, A$ é válido, temos $M \vDash A$. Como $B, \Pi \Rightarrow \Lambda$ é válido, temos $M \nvDash B$. Mas então $M \nvDash A \to B$, que é o que queríamos mostrar. □

Corolário 10.29. *Se $\vdash A$, então A é válido.*

Corolário 10.30. *Se $\Gamma \vdash A$, então $\Gamma \vDash A$.*

Prova. Se $\Gamma \vdash A$, então, para algum subconjunto finito $\Gamma_0 \subseteq \Gamma$, há uma derivação de $\Gamma_0 \Rightarrow A$. Por Teorema 10.28, toda estrutura M ou torna algum $B \in \Gamma_0$ falso ou torna A verdadeiro. Logo, se $M \vDash \Gamma$, então também $M \vDash A$. □

Corolário 10.31. *Se Γ é satisfatível, então é consistente.*

Prova. Provamos a contrapositiva. Suponha que Γ não seja consistente. Então há um $\Gamma_0 \subseteq \Gamma$ finito e uma derivação de $\Gamma_0 \Rightarrow$. Por Teorema 10.28, $\Gamma_0 \Rightarrow$ é válido. Em outras palavras, para toda estrutura M, há $C \in \Gamma_0$ tal que $M \nvDash C$, e como $\Gamma_0 \subseteq \Gamma$, esse C também está em Γ. Assim, nenhuma M satisfaz Γ, e Γ não é satisfatível. □

10.13 Derivações com Identidade

Derivações com identidade requerem sequentes iniciais e regras de inferência adicionais.

Definição 10.32 (Sequentes iniciais para =). Se t é um termo fechado, então $\Rightarrow t = t$ é um sequente inicial.

As regras para = são (t_1 e t_2 são termos fechados):

$$\frac{t_1 = t_2, \Gamma \Rightarrow \Delta, A(t_1)}{t_1 = t_2, \Gamma \Rightarrow \Delta, A(t_2)} = \qquad \frac{t_1 = t_2, \Gamma \Rightarrow \Delta, A(t_2)}{t_1 = t_2, \Gamma \Rightarrow \Delta, A(t_1)} =$$

Exemplo 10.33. Se s e t são termos fechados, então $s = t, A(s) \vdash A(t)$:

$$\dfrac{\dfrac{A(s) \Rightarrow A(s)}{s = t, A(s) \Rightarrow A(s)}\,\text{WL}}{s = t, A(s) \Rightarrow A(t)}\,=$$

Isso pode ser familiar como o princípio da substitutibilidade de idênticos, ou Lei de Leibniz.

LK prova que = é simétrica e transitiva:

$$\dfrac{\dfrac{\Rightarrow t_1 = t_1}{t_1 = t_2 \Rightarrow t_1 = t_1}\,\text{WL}}{t_1 = t_2 \Rightarrow t_2 = t_1}\,= \qquad \dfrac{\dfrac{\dfrac{t_1 = t_2 \Rightarrow t_1 = t_2}{t_2 = t_3, t_1 = t_2 \Rightarrow t_1 = t_2}\,\text{WL}}{t_2 = t_3, t_1 = t_2 \Rightarrow t_1 = t_3}\,=}{t_1 = t_2, t_2 = t_3 \Rightarrow t_1 = t_3}\,\text{XL}$$

Na derivação à esquerda, a fórmula $x = t_1$ é o nosso $A(x)$. À direita, tomamos $A(x)$ como $t_1 = x$.

10.14 Correção com Identidade

Proposição 10.34. LK *com sequentes iniciais e regras para a identidade é correto.*

Prova. Sequentes iniciais da forma $\Rightarrow t = t$ são válidos, pois para toda estrutura M, $M \vDash t = t$. (Note que supomos que o termo t seja fechado, isto é, que não contenha variáveis, de modo que atribuições de variáveis são irrelevantes).

Suponha que a última inferência em uma derivação seja =. Então a premissa é $t_1 = t_2, \Gamma \Rightarrow \Delta, A(t_1)$ e a conclusão é $t_1 = t_2, \Gamma \Rightarrow \Delta, A(t_2)$. Considere uma estrutura $\mathfrak{M}$. Precisamos mostrar que a conclusão é válida, isto é, se $\mathfrak{M} \vDash t_1 = t_2$ e $\mathfrak{M} \vDash \Gamma$, então ou $\mathfrak{M} \vDash C$ para algum $C \in \Delta$ ou $\mathfrak{M} \vDash A(t_2)$.

Pela hipótese de indução, a premissa é válida. Isso significa que, se $\mathfrak{M} \vDash t_1 = t_2$ e $\mathfrak{M} \vDash \Gamma$, então ou (a) para algum $C \in \Delta$, $\mathfrak{M} \vDash C$, ou (b) $\mathfrak{M} \vDash A(t_1)$. No caso (a), terminamos. Considere o caso (b). Seja s uma atribuição de variáveis com $s(x) = \mathrm{Val}^{\mathfrak{M}}(t_1)$. Por Proposição 7.18, $\mathfrak{M}, s \vDash A(t_1)$. Como $s \sim_x s$, por Proposição 7.23, $\mathfrak{M}, s \vDash A(x)$. Como $\mathfrak{M} \vDash t_1 = t_2$, temos $\mathrm{Val}^{\mathfrak{M}}(t_1) = \mathrm{Val}^{\mathfrak{M}}(t_2)$, e portanto $s(x) = \mathrm{Val}^{\mathfrak{M}}(t_2)$. Aplicando Proposição 7.23 novamente, temos também $\mathfrak{M}, s \vDash A(t_2)$. Por Proposição 7.18, $\mathfrak{M} \vDash A(t_2)$. □

Resumo

Os **sistemas de prova** fornecem métodos puramente sintáticos para caracterizar consequência e compatibilidade entre sentenças. O **cálculo de sequentes** é um desses sistemas de prova. Uma **derivação** nele consiste em uma árvore de sequentes (um sequente $\Gamma \Rightarrow \Delta$ consiste em duas sequências de fórmulas separadas por $\Rightarrow$). Os sequentes mais ao topo em uma derivação são **sequentes iniciais** da forma $A \Rightarrow A$. Todos os demais sequentes, para que a derivação seja correta, devem ser corretamente justificados por uma de várias **regras de inferência**. Elas vêm em pares; uma regra para operar no lado esquerdo e no lado direito de um sequente para cada conectivo e quantificador. Por exemplo, se um sequente $\Gamma \Rightarrow \Delta, A \to B$ é justificado pela regra $\to$R, o sequente precedente (a **premissa**) deve ser $A, \Gamma \Rightarrow \Delta, B$. Algumas regras também permitem que a ordem ou o número de sentenças em um sequente seja manipulado, por exemplo, a regra XR permite que duas fórmulas do lado direito de um sequente sejam trocadas.

Se há uma derivação do sequente $\Rightarrow A$, dizemos que A é um **teorema** e escrevemos $\vdash A$. Se há uma derivação de $\Gamma_0 \Rightarrow A$ em que todo B em Γ_0 está em Γ, dizemos que A é **derivável de** Γ e escrevemos $\Gamma \vdash A$. Se há uma derivação de $\Gamma_0 \Rightarrow$ em que todo B em Γ_0 está em Γ, dizemos que Γ é **inconsistente**; caso contrário, **consistente**. Essas noções estão inter-relacionadas; por exemplo, $\Gamma \vdash A$ sse $\Gamma \cup \{\neg A\}$ é inconsistente. Elas estão também relacionadas às noções semânticas correspondentes; por exemplo, se $\Gamma \vdash A$, então $\Gamma \vDash A$. Essa propriedade dos sistemas de prova—o que pode ser derivado de Γ tem garantia de ser consequência de Γ—é chamada de **correção**. O **teorema da correção** é provado por indução sobre o comprimento das derivações, mostrando que cada inferência individual preserva a validade do sequente conclusão, desde que os sequentes premissa sejam válidos.

Problemas

Problema 10.1. Dê derivações dos seguintes sequentes:

1. $A \wedge (B \wedge C) \Rightarrow (A \wedge B) \wedge C$.
2. $A \vee (B \vee C) \Rightarrow (A \vee B) \vee C$.
3. $A \to (B \to C) \Rightarrow B \to (A \to C)$.
4. $A \Rightarrow \neg\neg A$.

Problema 10.2. Dê derivações dos seguintes sequentes:

1. $(A \vee B) \to C \Rightarrow A \to C$.
2. $(A \to C) \wedge (B \to C) \Rightarrow (A \vee B) \to C$.
3. $\Rightarrow \neg(A \wedge \neg A)$.
4. $B \to A \Rightarrow \neg A \to \neg B$.
5. $\Rightarrow (A \to \neg A) \to \neg A$.

6. $\Rightarrow \neg(A \to B) \to \neg B$.
7. $A \to C \Rightarrow \neg(A \wedge \neg C)$.
8. $A \wedge \neg C \Rightarrow \neg(A \to C)$.
9. $A \vee B, \neg B \Rightarrow A$.
10. $\neg A \vee \neg B \Rightarrow \neg(A \wedge B)$.
11. $\Rightarrow (\neg A \wedge \neg B) \to \neg(A \vee B)$.
12. $\Rightarrow \neg(A \vee B) \to (\neg A \wedge \neg B)$.

Problema 10.3. Dê derivações dos seguintes sequentes:

1. $\neg(A \to B) \Rightarrow A$.
2. $\neg(A \wedge B) \Rightarrow \neg A \vee \neg B$.
3. $A \to B \Rightarrow \neg A \vee B$.
4. $\Rightarrow \neg\neg A \to A$.
5. $A \to B, \neg A \to B \Rightarrow B$.
6. $(A \wedge B) \to C \Rightarrow (A \to C) \vee (B \to C)$.
7. $(A \to B) \to A \Rightarrow A$.
8. $\Rightarrow (A \to B) \vee (B \to C)$.

(Todos esses requerem a regra CR.)

Problema 10.4. Dê derivações dos seguintes sequentes:

1. $\Rightarrow (\forall x\, A(x) \wedge \forall y\, B(y)) \to \forall z\, (A(z) \wedge B(z))$.
2. $\Rightarrow (\exists x\, A(x) \vee \exists y\, B(y)) \to \exists z\, (A(z) \vee B(z))$.
3. $\forall x\, (A(x) \to B) \Rightarrow \exists y\, A(y) \to B$.
4. $\forall x\, \neg A(x) \Rightarrow \neg\exists x\, A(x)$.

5. $\Rightarrow \neg\exists x\, A(x) \to \forall x\, \neg A(x)$.

6. $\Rightarrow \neg\exists x\, \forall y\, ((A(x,y) \to \neg A(y,y)) \wedge (\neg A(y,y) \to A(x,y)))$.

Problema 10.5. Dê derivações dos seguintes sequentes:

1. $\Rightarrow \neg\forall x\, A(x) \to \exists x\, \neg A(x)$.

2. $(\forall x\, A(x) \to B) \Rightarrow \exists y\, (A(y) \to B)$.

3. $\Rightarrow \exists x\, (A(x) \to \forall y\, A(y))$.

(Todos esses requerem a regra CR.)

Problema 10.6. Prove Proposição 10.16

Problema 10.7. Prove que $\Gamma \vdash \neg A$ se, e somente se, $\Gamma \cup \{A\}$ é inconsistente.

Problema 10.8. Complete a prova de Teorema 10.28.

Problema 10.9. Dê derivações dos seguintes sequentes:

1. $\Rightarrow \forall x\, \forall y\, ((x = y \wedge A(x)) \to A(y))$

2. $\exists x\, A(x) \wedge \forall y\, \forall z\, ((A(y) \wedge A(z)) \to y = z) \Rightarrow \exists x\, (A(x) \wedge \forall y\, (A(y) \to y = x))$

CAPÍTULO 11

Dedução Natural

11.1 Regras e Derivações

Sistemas de dedução natural pretendem ser um paralelo próximo do raciocínio informal usado em provas matemáticas (daí serem, de certo modo, "naturais"). Provas em dedução natural começam com hipóteses. Em seguida, aplicam-se regras de inferência. As hipóteses são "descarregadas" pelas regras de inferência ¬Intro, →Intro, ∨Elim e ∃Elim, e o rótulo da hipótese descarregada é colocado ao lado da inferência para maior clareza.

Definição 11.1 (Hipótese). Uma *hipótese* é qualquer sentença na posição mais alta de qualquer ramo.

Derivações em dedução natural são certas árvores de sentenças, em que as sentenças no topo são hipóteses e, se uma sentença está abaixo de um, dois ou três outros sequentes, ela deve seguir corretamente por uma regra de inferência. As sentenças no topo da inferência são chamadas de *premissas* e a sentença abaixo, de *conclusão* da inferência. As regras vêm em pares, uma regra de introdução e uma de eliminação para cada operador. Elas introduzem um operador na conclusão ou removem um operador de

uma premissa da regra. Algumas das regras permitem que uma hipótese de certo tipo seja *descarregada*. Para indicar qual hipótese é descarregada por qual inferência, também atribuímos rótulos tanto à hipótese quanto à inferência. Isso é indicado escrevendo-se a hipótese como "$[A]^n$."

É costume considerar regras para todos os operadores $\wedge$, $\vee$, $\rightarrow$, $\neg$ e $\bot$, mesmo que alguns deles sejam definidos.

11.2 Regras Proposicionais

Regras para $\wedge$

$$\frac{A \qquad B}{A \wedge B}\ \wedge\text{Intro}$$

$$\frac{A \wedge B}{A}\ \wedge\text{Elim} \qquad \frac{A \wedge B}{B}\ \wedge\text{Elim}$$

Regras para $\vee$

$$\frac{A}{A \vee B}\ \vee\text{Intro} \qquad \frac{B}{A \vee B}\ \vee\text{Intro}$$

$$n\ \frac{A \vee B \qquad \begin{array}{c}[A]^n \\ \vdots \\ C\end{array} \qquad \begin{array}{c}[B]^n \\ \vdots \\ C\end{array}}{C}\ \vee\text{Elim}$$

Regras para $\rightarrow$

$$n\ \frac{\begin{array}{c}[A]^n \\ \vdots \\ B\end{array}}{A \rightarrow B}\ \rightarrow\text{Intro}$$

$$\frac{A \rightarrow B \qquad A}{B}\ \rightarrow\text{Elim}$$

Regras para $\neg$

$$\dfrac{\begin{array}{c}[A]^n\\ \vdots\\ \bot\end{array}}{\neg A}\ {}^n\neg\text{Intro} \qquad\qquad \dfrac{\neg A \quad A}{\bot}\ \neg\text{Elim}$$

Regras para $\bot$

$$\dfrac{\bot}{A}\ \bot_I \qquad\qquad \dfrac{\begin{array}{c}[\neg A]^n\\ \vdots\\ \bot\end{array}}{A}\ {}^n\bot_C$$

Note que ¬Intro e $\bot_C$ são muito semelhantes: a diferença é que ¬Intro deriva uma sentença negada $\neg A$, mas $\bot_C$, uma sentença positiva A.

Sempre que uma regra indica que alguma hipótese pode ser descarregada, nós tomamos isso como uma permissão, mas não uma exigência. Por exemplo, na regra →Intro, podemos descarregar qualquer número de hipóteses da forma A na derivação da premissa B, inclusive zero.

11.3 Regras de Quantificadores

Regras para $\forall$

$$\dfrac{A(a)}{\forall x\, A(x)}\ \forall\text{Intro} \qquad\qquad \dfrac{\forall x\, A(x)}{A(t)}\ \forall\text{Elim}$$

Nas regras para $\forall$, t é um termo fechado (um termo que não contém nenhuma variável), e a é um símbolo de constante que não ocorre na conclusão $\forall x\, A(x)$, nem em qualquer hipótese que esteja não descarregada na derivação que termina com

a premissa $A(a)$. Chamamos a de *variável própria* da inferência ∀Intro.[1]

Regras para ∃

$$\frac{A(t)}{\exists x\, A(x)}\ \exists\text{Intro} \qquad\qquad \begin{array}{c} [A(a)]^n \\ \vdots \\ \exists x\, A(x) \qquad C \\ \hline C \end{array}\ n\ \exists\text{Elim}$$

Novamente, t é um termo fechado, e a é um símbolo de constante que não ocorre na premissa $\exists x\, A(x)$, na conclusão C, nem em qualquer hipótese que esteja não descarregada nas derivações que terminam com as duas premissas (além das hipóteses $A(a)$). Chamamos a de *variável própria* da inferência ∃Elim.

A condição de que uma variável própria não ocorra nem nas premissas nem em qualquer hipótese que esteja não descarregada nas derivações que levam às premissas para a inferência ∀Intro ou ∃Elim é chamada de *condição de variável própria.*

Lembre-se da convenção de que, quando A é uma fórmula com a variável x livre, indicamos isso escrevendo $A(x)$. No mesmo contexto, $A(t)$ é então uma abreviação de $A[t/x]$. Assim, poderíamos também escrever a regra ∃Intro como:

$$\frac{A[t/x]}{\exists x\, A}\ \exists\text{Intro}$$

Note que t pode já ocorrer em A, por exemplo, A poderia ser $P(t,x)$. Assim, inferir $\exists x\, P(t,x)$ de $P(t,t)$ é uma aplicação correta de ∃Intro—pode-se "substituir" uma ou mais, e não necessariamente todas, as ocorrências de t na premissa pela variável x ligada. Contudo, as condições de variável própria em ∀Intro e ∃Elim exigem que a símbolo de constante a não ocorra em A.

[1] Usamos o termo "variável própria" embora a na regra acima seja uma constante. Isso tem razões históricas.

Portanto, não se pode inferir corretamente $\forall x\, P(a,x)$ de $P(a,a)$ usando $\forall$Intro.

Em $\exists$Intro e $\forall$Elim não há restrições, e o termo t pode ser qualquer coisa, de modo que não temos de nos preocupar com nenhuma condição. Por outro lado, nas regras $\exists$Elim e $\forall$Intro, a condição de variável própria exige que a símbolo de constante a não ocorra em lugar algum na conclusão nem em uma hipótese não descarregada. A condição é necessária para garantir que o sistema seja correto, isto é, que só derivam sentenças de hipóteses não descarregadas das quais elas se seguem. Sem essa condição, o seguinte seria permitido:

$$\dfrac{\exists x\, A(x) \qquad \dfrac{[A(a)]^1}{\forall x\, A(x)}\,{}^{*}\forall\text{Intro}}{\forall x\, A(x)}\,\exists\text{Elim}$$

Contudo, $\exists x\, A(x) \nvdash \forall x\, A(x)$.

Como as regras de eliminação para quantificadores só permitem substituir variáveis por termos fechados, segue-se que qualquer fórmula que possa ser derivada de um conjunto de sentenças é ela própria uma sentença.

11.4 Derivações

Já dissemos o que é uma hipótese e demos as regras de inferência. Derivações em dedução natural são geradas indutivamente a partir disso: cada derivação ou é uma hipótese por si só, ou consiste em uma, duas ou três derivações seguidas de uma inferência correta.

Definição 11.2 (Derivação). Uma *derivação* de uma sentença A a partir de hipóteses Γ é uma árvore finita de sentenças que satisfaz as seguintes condições:

1. As sentenças no topo da árvore ou estão em Γ ou são descarregadas por uma inferência na árvore.

2. A sentença na base da árvore é A.

3. Toda sentença na árvore exceto a sentença A na base é uma premissa de uma aplicação correta de uma regra de inferência cuja conclusão está diretamente abaixo daquela sentença na árvore.

Dizemos então que A é a *conclusão* da derivação e Γ, suas hipóteses não descarregadas.

Se existe uma derivação de A a partir de Γ, dizemos que A é *derivável* a partir de Γ, ou, em símbolos: $\Gamma \vdash A$. Se há uma derivação de A em que toda hipótese é descarregada, escrevemos $\vdash A$.

Exemplo 11.3. Toda hipótese por si só é uma derivação. Assim, por exemplo, A por si só é uma derivação, e B por si só também. Podemos obter uma nova derivação a partir delas aplicando, digamos, a regra $\wedge$Intro,

$$\frac{A \qquad B}{A \wedge B} \wedge\text{Intro}$$

Essas regras devem ser gerais: podemos substituir o A e B nela por quaisquer sentenças, por exemplo, por C e D. Então a conclusão seria $C \wedge D$, e assim

$$\frac{C \qquad D}{C \wedge D} \wedge\text{Intro}$$

é uma derivação correta. É claro que também podemos trocar as hipóteses, de modo que D desempenhe o papel de A e C o de B. Assim,

$$\frac{D \qquad C}{D \wedge C} \wedge\text{Intro}$$

também é uma derivação correta.

Podemos agora aplicar outra regra, digamos, $\rightarrow$Intro, que nos permite concluir um condicional e nos permite descarregar qualquer hipótese que seja idêntica ao antecedente desse condicional. Assim, ambas as seguintes seriam derivações corretas:

$$1\,\dfrac{\dfrac{[C]^1 \qquad D}{C \wedge D}\,\wedge\text{Intro}}{C \to (C \wedge D)}\,\to\text{Intro} \qquad 1\,\dfrac{\dfrac{C \qquad [D]^1}{C \wedge D}\,\wedge\text{Intro}}{D \to (C \wedge D)}\,\to\text{Intro}$$

Elas mostram, respectivamente, que $D \vdash C \to (C \wedge D)$ e $C \vdash D \to (C \wedge D)$.

Lembre-se de que descarregar hipóteses é uma permissão, não uma exigência: não temos de descarregar as hipóteses. Em particular, podemos aplicar uma regra mesmo que as hipóteses não estejam presentes na derivação. Por exemplo, o seguinte é válido, embora não haja hipótese A a ser descarregada:

$$1\,\dfrac{B}{A \to B}\,\to\text{Intro}$$

11.5 Exemplos de Derivações

Exemplo 11.4. Vamos dar uma derivação da sentença $(A \wedge B) \to A$.

Começamos escrevendo a conclusão desejada na parte inferior da derivação.

$$\dfrac{}{(A \wedge B) \to A}$$

Em seguida, precisamos descobrir que tipo de inferência poderia resultar em uma sentença desta forma. O operador principal da conclusão é $\to$, então tentaremos chegar à conclusão usando a regra $\to$Intro. O melhor é anotar as suposições envolvidas e rotular as regras de inferência conforme avançamos, para que seja fácil ver se todas as suposições foram descarregadas ao final da prova.

$$1\,\dfrac{\begin{array}{c}[A \wedge B]^1 \\ \vdots \\ A\end{array}}{(A \wedge B) \to A}\,\to\text{Intro}$$

Agora precisamos preencher os passos da suposição $A \wedge B$ até A. Como só temos um conectivo com o qual lidar, $\wedge$, devemos usar a regra $\wedge$ elim. Isso nos dá a seguinte prova:

$$\dfrac{\dfrac{[A \wedge B]^1}{A}\,\wedge\text{Elim}}{(A \wedge B) \to A}\,{}^1\!\to\text{Intro}$$

Agora temos uma derivação correta de $(A \wedge B) \to A$.

Exemplo 11.5. Agora vamos dar uma derivação de $(\neg A \vee B) \to (A \to B)$.

Começamos escrevendo a conclusão desejada na parte inferior da derivação.

$$\dfrac{}{(\neg A \vee B) \to (A \to B)}$$

Para encontrar uma regra lógica que possa nos dar esta conclusão, observamos os conectivos lógicos na conclusão: $\neg$, $\vee$ e $\to$. Por enquanto, só nos interessa a primeira ocorrência de $\to$, porque é o operador principal da sentença na conclusão, enquanto $\neg$, $\vee$ e a segunda ocorrência de $\to$ estão dentro do escopo de outro conectivo, então cuidaremos deles depois. Começamos, portanto, com a regra $\to$Intro. Uma aplicação correta deve ser assim:

$$\dfrac{\begin{array}{c}[\neg A \vee B]^1\\ \vdots\\ A \to B\end{array}}{(\neg A \vee B) \to (A \to B)}\,{}^1\!\to\text{Intro}$$

Isso nos deixa com duas possibilidades para continuar. Podemos continuar trabalhando de baixo para cima e procurar outra aplicação da regra $\to$Intro, ou podemos trabalhar de cima para baixo e aplicar a regra $\vee$Elim. Vamos adotar a segunda opção. Usaremos a suposição $\neg A \vee B$ como a premissa mais à esquerda de $\vee$Elim. Para uma aplicação válida de $\vee$Elim, as outras duas

premissas devem ser idênticas à conclusão $A \to B$, mas cada uma pode ser derivada, por sua vez, de outra suposição, a saber, um dos dois disjuntos de $\neg A \vee B$. Nossa derivação ficará, então, assim:

$$\dfrac{\dfrac{[\neg A \vee B]^1 \qquad \begin{matrix}[\neg A]^2 \\ \vdots \\ A \to B\end{matrix} \qquad \begin{matrix}[B]^2 \\ \vdots \\ A \to B\end{matrix}}{A \to B}\,\vee\text{Elim}\,(2)}{(\neg A \vee B) \to (A \to B)}\,\to\text{Intro}\,(1)$$

Em cada um dos dois ramos à direita, queremos derivar $A \to B$, o que é melhor feito usando $\to$Intro.

$$\dfrac{\dfrac{[\neg A \vee B]^1 \qquad \dfrac{\begin{matrix}[\neg A]^2, [A]^3 \\ \vdots \\ B\end{matrix}}{A \to B}\,\to\text{Intro}\,(3) \qquad \dfrac{\begin{matrix}[B]^2, [A]^4 \\ \vdots \\ B\end{matrix}}{A \to B}\,\to\text{Intro}\,(4)}{A \to B}\,\vee\text{Elim}\,(2)}{(\neg A \vee B) \to (A \to B)}\,\to\text{Intro}\,(1)$$

Nas duas partes que faltam na derivação, precisamos de derivações de B a partir de $\neg A$ e A no meio, e a partir de A e B à esquerda. Comecemos pelo primeiro caso. $\neg A$ e A são as duas premissas de $\neg$Elim:

$$\dfrac{\begin{matrix}\dfrac{[\neg A]^2 \qquad [A]^3}{\bot}\,\neg\text{Elim} \\ \vdots \\ B\end{matrix}}{}$$

Usando $\bot_I$, podemos obter B como conclusão e completar o ramo.

$$\dfrac{\dfrac{}{\neg A \vee B}^{2} \quad \dfrac{\dfrac{\dfrac{\dfrac{[\neg A]^{2} \quad [A]^{3}}{\bot}\,\bot\text{Intro}}{B}\,\bot_I}{A \to B}^{3}\,{\to}\text{Intro} \quad \dfrac{\dfrac{[B]^{2}, [A]^{4}}{B}}{A \to B}^{4}\,{\to}\text{Intro}}{A \to B}\,{\vee}\text{Elim}}{(\neg A \vee B) \to (A \to B)}^{1}\,{\to}\text{Intro}$$

Vejamos agora o ramo mais à direita. Aqui é importante perceber que a definição de derivação *permite que hipóteses sejam descarregadas* mas *não exige* que o sejam. Em outras palavras, se pudermos derivar B a partir de uma das suposições A e B sem usar a outra, tudo bem. E para derivar B a partir de B é trivial: B por si só é tal uma derivação, e nenhuma inferência é necessária. Portanto, podemos simplesmente eliminar a suposição A.

$$\dfrac{\dfrac{}{\neg A \vee B}^{2} \quad \dfrac{\dfrac{\dfrac{\dfrac{[\neg A]^{2} \quad [A]^{3}}{\bot}\,\neg\text{Elim}}{B}\,\bot_I}{A \to B}^{3}\,{\to}\text{Intro} \quad \dfrac{[B]^{2}}{A \to B}\,{\to}\text{Intro}}{A \to B}\,{\vee}\text{Elim}}{(\neg A \vee B) \to (A \to B)}^{1}\,{\to}\text{Intro}$$

Observe que, na derivação concluída, a inferência $\to$Intro mais à direita na verdade não descarrega nenhuma hipótese.

Exemplo 11.6. Até agora não precisamos da regra $\bot_C$. Ela é especial porque nos permite descarregar uma hipótese que não é uma sub-fórmula da conclusão da regra. Está intimamente relacionada à regra $\bot_I$. Na verdade, a regra $\bot_I$ é um caso especial da regra $\bot_C$—existe uma lógica chamada "lógica intuicionista" na qual apenas $\bot_I$ é permitida. A regra $\bot_C$ é um último recurso quando nada mais funciona. Por exemplo, suponha que queiramos derivar $A \vee \neg A$. Nossa estratégia usual seria tentar derivar $A \vee \neg A$ usando $\vee$Intro. Mas isso exigiria derivar A ou $\neg A$ a partir de nenhuma suposição, e isso não pode ser feito. $\bot_C$ ao resgate!

$$\dfrac{\begin{array}{c}[\neg(A \vee \neg A)]^1 \\ \vdots \\ \bot\end{array}}{A \vee \neg A}\ 1\ \bot_C$$

Agora procuramos uma derivação de $\bot$ a partir de $\neg(A \vee \neg A)$. Como $\bot$ é a conclusão de $\neg$Elim, podemos tentar isso:

$$\dfrac{\dfrac{\begin{array}{c}[\neg(A \vee \neg A)]^1 \\ \vdots \\ \neg A\end{array} \qquad \begin{array}{c}[\neg(A \vee \neg A)]^1 \\ \vdots \\ A\end{array}}{\dfrac{\bot}{A \vee \neg A}\ 1\ \bot_C}\ \neg\text{Elim}$$

Nossa estratégia para encontrar uma derivação de $\neg A$ exige uma aplicação de $\neg$Intro:

$$\dfrac{\dfrac{\begin{array}{c}[\neg(A \vee \neg A)]^1, [A]^2 \\ \vdots \\ \bot\end{array}}{\neg A}\ 2\ \neg\text{Intro} \qquad \begin{array}{c}[\neg(A \vee \neg A)]^1 \\ \vdots \\ A\end{array}}{\dfrac{\bot}{A \vee \neg A}\ 1\ \bot_C}\ \neg\text{Elim}$$

Aqui, podemos obter $\bot$ facilmente aplicando $\neg$Elim à suposição $\neg(A \vee \neg A)$ e a $A \vee \neg A$, que segue da nossa nova suposição A por $\vee$Intro:

$$\dfrac{\dfrac{\dfrac{[\neg(A \vee \neg A)]^1 \qquad \dfrac{[A]^2}{A \vee \neg A}\ \vee\text{Intro}}{\bot}\ \neg\text{Elim}}{\neg A}\ 2\ \neg\text{Intro} \qquad \begin{array}{c}[\neg(A \vee \neg A)]^1 \\ \vdots \\ A\end{array}}{\dfrac{\bot}{A \vee \neg A}\ 1\ \bot_C}\ \neg\text{Elim}$$

No lado direito usamos a mesma estratégia, exceto que obtemos A por $\bot_C$:

$$\dfrac{\dfrac{\dfrac{[\neg(A \vee \neg A)]^1 \qquad \dfrac{[A]^2}{A \vee \neg A}\,\vee\text{Intro}}{\bot}\,\neg\text{Elim}}{\neg A}\,2\,\neg\text{Intro} \qquad \dfrac{\dfrac{[\neg(A \vee \neg A)]^1 \qquad \dfrac{[\neg A]^3}{A \vee \neg A}\,\vee\text{Intr}}{\bot}\,\neg\text{Elim}}{A}\,3\,\bot_C}{\bot}\,\neg\text{Elim}}{A \vee \neg A}\,1\,\bot_C$$

11.6 Derivações com Quantificadores

Exemplo 11.7. Ao lidar com quantificadores, temos de cuidar para não violar a condição de variável própria, e às vezes isso exige que brinquemos com a ordem de execução de certas inferências. Em geral, ajuda tentar cuidar primeiro das regras sujeitas à condição de variável própria (elas ficarão mais abaixo na prova finalizada).

Vamos ver como daríamos uma derivação de fórmula $\exists x\, \neg A(x) \to \neg \forall x\, A(x)$. Começando como de costume, escrevemos

$$\dfrac{}{\exists x\, \neg A(x) \to \neg \forall x\, A(x)}$$

Começamos escrevendo o que seria necessário para justificar esse último passo usando a regra $\to$Intro.

$$\dfrac{\begin{array}{c}[\exists x\, \neg A(x)]^1 \\ \vdots \\ \neg \forall x\, A(x)\end{array}}{\exists x\, \neg A(x) \to \neg \forall x\, A(x)}\,1\,\to\text{Intro}$$

Como não há uma regra óbvia para aplicar a $\neg \forall x\, A(x)$, prosseguiremos montando a derivação de modo que possamos usar a regra $\exists$Elim. Aqui devemos prestar atenção à condição de variável própria e escolher uma constante que não apareça em $\exists x\, A(x)$ nem em quaisquer suposições das quais ela dependa. (Como, no entanto, nenhum símbolo de constante aparece, qualquer escolha servirá.)

$$\dfrac{\dfrac{[\exists x\, \neg A(x)]^1 \qquad \neg\forall x\, A(x)}{\neg\forall x\, A(x)}\,\exists\text{Elim}^{\,2}}{\exists x\, \neg A(x) \to \neg\forall x\, A(x)}\,{\to}\text{Intro}^{\,1}$$

Para derivar $\neg\forall x\, A(x)$, tentaremos usar a regra ¬Intro: isso exige que derivemos uma contradição, possivelmente usando $\forall x\, A(x)$ como suposição adicional. É claro que essa contradição pode envolver a suposição $\neg A(a)$, que será descarregada pela inferência ∃Elim. Podemos montá-la da seguinte forma:

$$\dfrac{\dfrac{[\exists x\, \neg A(x)]^1 \qquad \dfrac{\bot}{\neg\forall x\, A(x)}\,\neg\text{Intro}^{\,3}}{\neg\forall x\, A(x)}\,\exists\text{Elim}^{\,2}}{\exists x\, \neg A(x) \to \neg\forall x\, A(x)}\,{\to}\text{Intro}^{\,1}$$

Parece que estamos perto de obter uma contradição. A regra mais fácil de aplicar é a ∀Elim, que não possui condição de variável própria. Como podemos usar qualquer termo que queiramos para substituir o x universalmente quantificado, faz mais sentido continuar usando a para chegarmos a uma contradição.

$$\dfrac{\dfrac{[\exists x\, \neg A(x)]^1 \qquad \dfrac{\dfrac{\bot}{\neg\forall x\, A(x)}\,\neg\text{Intro}^{\,3}}{\neg\forall x\, A(x)}\,\exists\text{Elim}^{\,2}}{\exists x\, \neg A(x) \to \neg\forall x\, A(x)}\,{\to}\text{Intro}^{\,1} \qquad \dfrac{[\neg A(a)]^2 \qquad \dfrac{[\forall x\, A(x)]^3}{A(a)}\,\forall\text{Elim}}{\bot}\,\neg\text{Elim}$$

É importante, especialmente ao lidar com quantificadores, verificar com cuidado neste ponto que a condição de variável própria não foi violada. Como a única regra que aplicamos sujeita

à condição de variável própria foi ∃Elim, e a variável própria a não ocorre em nenhuma suposição da qual dependa, esta é uma derivação correta.

Exemplo 11.8. Às vezes podemos derivar uma fórmula a partir de outras fórmulas. Nesses casos, podemos ter suposições não descarregadas. É importante acompanhar nossas suposições, bem como o objetivo final.

Vamos ver como daríamos uma derivação de fórmula $\exists x\, C(x,b)$ a partir das suposições $\exists x\, (A(x) \wedge B(x))$ e $\forall x\, (B(x) \to C(x,b))$. Começando como de costume, escrevemos a conclusão na parte inferior.

$$\dfrac{}{\exists x\, C(x,b)}$$

Temos duas premissas com as quais trabalhar. Para usar a primeira, isto é, tentar encontrar uma derivação de $\exists x\, C(x,b)$ a partir de $\exists x\, (A(x) \wedge B(x))$, usaríamos a regra ∃Elim. Como ela possui condição de variável própria, aplicaremos essa regra primeiro. Obtemos o seguinte:

$$\dfrac{\exists x\, (A(x) \wedge B(x)) \qquad \begin{array}{c}[A(a) \wedge B(a)]^1 \\ \vdots \\ \exists x\, C(x,b)\end{array}}{\exists x\, C(x,b)}\, 1\ \exists\text{Elim}$$

As duas suposições com as quais estamos trabalhando compartilham B. Pode ser útil neste ponto aplicar ∧Elim para separar $B(a)$.

$$\dfrac{\exists x\, (A(x) \wedge B(x)) \qquad \begin{array}{c}\dfrac{[A(a) \wedge B(a)]^1}{B(a)}\ \wedge\text{Elim} \\ \vdots \\ \exists x\, C(x,b)\end{array}}{\exists x\, C(x,b)}\, 1\ \exists\text{Elim}$$

A segunda suposição com a qual temos de trabalhar é $\forall x\,(B(x) \to C(x,b))$. Como não há condição de variável própria, podemos instanciar x com o símbolo de constante a usando $\forall$Elim para obter $B(a) \to C(a,b)$. Agora temos tanto $B(a) \to C(a,b)$ quanto $B(a)$. Nosso próximo passo deve ser uma aplicação direta da regra $\to$Elim.

$$\dfrac{\dfrac{\dfrac{\forall x\,(B(x) \to C(x,b))}{B(a) \to C(a,b)}\,\forall\text{Elim} \qquad \dfrac{[A(a) \wedge B(a)]^1}{B(a)}\,\wedge\text{Elim}}{C(a,b)}\,\to\text{Elim}}{\vdots}$$

$$1\ \dfrac{\exists x\,(A(x) \wedge B(x)) \qquad \exists x\,C(x,b)}{\exists x\,C(x,b)}\,\exists\text{Elim}$$

Estamos quase lá! Uma aplicação de $\exists$Intro e atingimos nosso objetivo.

$$\dfrac{\dfrac{\dfrac{\forall x\,(B(x) \to C(x,b))}{B(a) \to C(a,b)}\,\forall\text{Elim} \qquad \dfrac{[A(a) \wedge B(a)]^1}{B(a)}\,\wedge\text{Elim}}{C(a,b)}\,\to\text{Elim}}{\exists x\,C(x,b)}\,\exists\text{Intro}$$

$$1\ \dfrac{\exists x\,(A(x) \wedge B(x)) \qquad \exists x\,C(x,b)}{\exists x\,C(x,b)}\,\exists\text{Elim}$$

Como garantimos em cada etapa que as condições de variável própria não foram violadas, podemos estar confiantes de que esta é uma derivação correta.

Exemplo 11.9. Dê uma derivação de fórmula $\neg\forall x\,A(x)$ a partir das suposições $\forall x\,A(x) \to \exists y\,B(y)$ e $\neg\exists y\,B(y)$. Começando como de costume, escrevemos a fórmula alvo na parte inferior.

$$\dfrac{}{\neg\forall x\,A(x)}$$

A última linha da derivação é uma negação, então vamos tentar usar $\neg$Intro. Isso exigirá que descubramos como derivar uma contradição.

$$\dfrac{\dfrac{[\forall x\, A(x)]^1}{\vdots}}{\dfrac{\bot}{\neg\forall x\, A(x)}\,1}\,\neg\text{Intro}$$

Até aqui, tudo bem. Podemos usar ∀Elim, mas não é óbvio se isso nos ajudará a alcançar nosso objetivo. Em vez disso, vamos usar uma de nossas suposições. $\forall x\, A(x) \to \exists y\, B(y)$ junto com $\forall x\, A(x)$ nos permitirá usar a regra →Elim.

$$\dfrac{\dfrac{\dfrac{\forall x\, A(x) \to \exists y\, B(y) \qquad [\forall x\, A(x)]^1}{\exists y\, B(y)}\,\to\text{Elim}}{\vdots}}{\dfrac{\bot}{\neg\forall x\, A(x)}\,1}\,\neg\text{Intro}$$

Agora temos uma suposição final com a qual trabalhar, e parece que isso nos ajudará a chegar a uma contradição usando ¬Elim.

$$\dfrac{\dfrac{\neg\exists y\, B(y) \qquad \dfrac{\forall x\, A(x) \to \exists y\, B(y) \qquad [\forall x\, A(x)]^1}{\exists y\, B(y)}\,\to\text{Elim}}{\bot}\,\neg\text{Elim}}{\dfrac{\bot}{\neg\forall x\, A(x)}\,1}\,\neg\text{Intro}$$

11.7 Noções da Teoria da Prova

Assim como definimos várias noções semânticas importantes (validade, consequência, satisfatibilidade), agora definimos noções *da Teoria da Prova* correspondentes. Elas não são definidas por apelo à satisfação de sentenças em estruturas, mas por apelo à derivabilidade ou não derivabilidade de certas sentenças a partir de outras. Foi uma descoberta importante que essas noções coincidem. Que coincidem é o conteúdo dos teoremas de *correção* e *completude*.

Definição 11.10 (Teoremas). Uma sentença A é um *teorema* se há uma derivação de A em dedução natural na qual todas as hipóteses são descarregadas. Escrevemos $\vdash A$ se A é um teorema e $\nvdash A$ se não o é.

Definição 11.11 (Derivabilidade). Uma sentença A é *derivável de* um conjunto de sentenças Γ, $\Gamma \vdash A$, se há uma derivação com conclusão A e na qual toda hipótese é ou descarregada ou está em Γ. Se A não é derivável de Γ, escrevemos $\Gamma \nvdash A$.

Definição 11.12 (Consistência). Um conjunto de sentenças Γ é *inconsistente* se, e somente se, $\Gamma \vdash \bot$. Se Γ não é inconsistente, isto é, se $\Gamma \nvdash \bot$, dizemos que é *consistente.*

Proposição 11.13 (Reflexividade). *Se $A \in \Gamma$, então $\Gamma \vdash A$.*

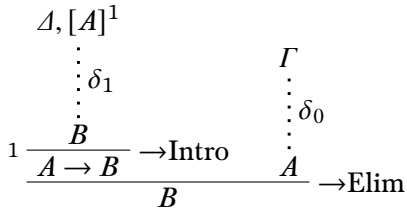
Prova. A hipótese A por si só é uma derivação de A em que toda hipótese não descarregada (isto é, A) está em Γ. □

Proposição 11.14 (Monotonicidade). *Se $\Gamma \subseteq \Delta$ e $\Gamma \vdash A$, então $\Delta \vdash A$.*

Prova. Qualquer derivação de A a partir de Γ é também uma derivação de A a partir de Δ. □

Proposição 11.15 (Transitividade). *Se $\Gamma \vdash A$ e $\{A\} \cup \Delta \vdash B$, então $\Gamma \cup \Delta \vdash B$.*

Prova. Se $\Gamma \vdash A$, há uma derivação δ_0 de A com todas as hipóteses não descarregadas em Γ. Se $\{A\} \cup \Delta \vdash B$, então há uma derivação δ_1 de B com todas as hipóteses não descarregadas em $\{A\} \cup \Delta$. Considere agora:



As hipóteses não descarregadas estão agora todas entre $\Gamma \cup \Delta$, de modo que isto mostra $\Gamma \cup \Delta \vdash B$. □

Quando $\Gamma = \{A_1, A_2, \ldots, A_k\}$ é um conjunto finito, podemos usar a notação simplificada $A_1, A_2, \ldots, A_k \vdash B$ para $\Gamma \vdash B$; em particular, $A \vdash B$ significa que $\{A\} \vdash B$.

Note que se $\Gamma \vdash A$ e $A \vdash B$, então $\Gamma \vdash B$. Segue também que se $A_1, \ldots, A_n \vdash B$ e $\Gamma \vdash A_i$ para cada i, então $\Gamma \vdash B$.

Proposição 11.16. *As seguintes afirmações são equivalentes.*

1. *Γ é inconsistente.*
2. *$\Gamma \vdash A$ para toda sentença A.*
3. *$\Gamma \vdash A$ e $\Gamma \vdash \neg A$ para alguma sentença A.*

Prova. Exercício. □

Proposição 11.17 (Compacidade). 1. *Se $\Gamma \vdash A$, então há um subconjunto finito $\Gamma_0 \subseteq \Gamma$ tal que $\Gamma_0 \vdash A$.*

2. *Se todo subconjunto finito de Γ é consistente, então Γ é consistente.*

Prova. 1. Se $\Gamma \vdash A$, então há uma derivação δ de A a partir de Γ. Seja Γ_0 o conjunto de hipóteses não descarregadas de δ. Como toda derivação é finita, Γ_0 só pode conter finitamente muitas sentenças. Assim, δ é uma derivação de A a partir de um Γ_0 finito $\subseteq \Gamma$.

2. Isto é a contrapositiva de (1) no caso especial $A \equiv \bot$. □

11.8 Derivabilidade e Consistência

Estabeleceremos agora várias propriedades da relação de derivabilidade. Elas são independentemente interessantes, mas cada uma desempenhará um papel na prova do teorema da completude.

Proposição 11.18. *Se $\Gamma \vdash A$ e $\Gamma \cup \{A\}$ é inconsistente, então Γ é inconsistente.*

Prova. Seja a derivação de A a partir de Γ igual a δ_1 e a derivação de $\bot$ a partir de $\Gamma \cup \{A\}$ igual a δ_2. Podemos então derivar:

$$\dfrac{\dfrac{\begin{array}{c}\Gamma, [A]^1 \\ \vdots\,\delta_2 \\ \bot\end{array}}{\neg A}\,1\,\neg\text{Intro} \qquad \begin{array}{c}\Gamma \\ \vdots\,\delta_1 \\ A\end{array}}{\bot}\,\neg\text{Elim}$$

Na nova derivação, a hipótese A é descarregada, de modo que é uma derivação a partir de Γ. □

Proposição 11.19. *$\Gamma \vdash A$ se, e somente se, $\Gamma \cup \{\neg A\}$ é inconsistente.*

Prova. Primeiro suponha que $\Gamma \vdash A$, isto é, há uma derivação δ_0 de A a partir de hipóteses não descarregadas Γ. Obtemos uma derivação de $\bot$ a partir de $\Gamma \cup \{\neg A\}$ da seguinte forma:

$$\dfrac{\neg A \qquad \begin{array}{c}\Gamma \\ \vdots\,\delta_0 \\ A\end{array}}{\bot}\,\neg\text{Elim}$$

Agora suponha que $\Gamma \cup \{\neg A\}$ é inconsistente, e seja δ_1 a derivação correspondente de $\bot$ a partir de hipóteses não descarregadas em $\Gamma \cup \{\neg A\}$. Obtemos uma derivação de A apenas a partir de Γ usando $\bot_C$:

$$\dfrac{\begin{array}{c}\Gamma, [\neg A]^1 \\ \vdots \\ \delta_1 \\ \vdots \\ \bot\end{array}}{A}\ 1\ \bot_C$$

□

Proposição 11.20. *Se $\Gamma \vdash A$ e $\neg A \in \Gamma$, então Γ é inconsistente.*

Prova. Suponha que $\Gamma \vdash A$ e $\neg A \in \Gamma$. Então há uma derivação δ de A a partir de Γ. Considere esta aplicação simples da regra ¬Elim:

$$\dfrac{\neg A \qquad \begin{array}{c}\Gamma \\ \vdots \\ \delta \\ \vdots \\ A\end{array}}{\bot}\ \neg\text{Elim}$$

Como $\neg A \in \Gamma$, todas as hipóteses não descarregadas estão em Γ; isto mostra que $\Gamma \vdash \bot$. □

Proposição 11.21. *Se $\Gamma \cup \{A\}$ e $\Gamma \cup \{\neg A\}$ são ambos inconsistentes, então Γ é inconsistente.*

Prova. Há derivações δ_1 e δ_2 de $\bot$ a partir de $\Gamma \cup \{A\}$ e de $\bot$ a partir de $\Gamma \cup \{\neg A\}$, respectivamente. Podemos então derivar

$$\dfrac{\dfrac{\begin{array}{c}\Gamma, [\neg A]^2 \\ \vdots \\ \delta_2 \\ \vdots \\ \bot\end{array}}{\neg\neg A}\ 2\ \neg\text{Intro} \qquad \dfrac{\begin{array}{c}\Gamma, [A]^1 \\ \vdots \\ \delta_1 \\ \vdots \\ \bot\end{array}}{\neg A}\ 1\ \neg\text{Intro}}{\bot}\ \neg\text{Elim}$$

Como as hipóteses A e $\neg A$ são descarregadas, isto é uma derivação de $\bot$ apenas a partir de Γ. Logo Γ é inconsistente. □

11.9 Derivabilidade e os Conectivos Proposicionais

Estabelecemos que a relação de derivabilidade $\vdash$ da dedução natural é forte o suficiente para estabelecer alguns fatos básicos envolvendo os conectivos proposicionais, tais como $A \wedge B \vdash A$ e $A, A \to B \vdash B$ (modus ponens). Esses fatos são necessários para a prova do teorema da completude.

Proposição 11.22. *1. Tanto $A \wedge B \vdash A$ quanto $A \wedge B \vdash B$.*

2. $A, B \vdash A \wedge B$.

Prova. 1. Podemos derivar ambos:

$$\dfrac{A \wedge B}{A}\,\wedge\text{Elim} \qquad \dfrac{A \wedge B}{B}\,\wedge\text{Elim}$$

2. Podemos derivar:

$$\dfrac{A \qquad B}{A \wedge B}\,\wedge\text{Intro}$$

□

Proposição 11.23. *1. $A \vee B, \neg A, \neg B$ é inconsistente.*

2. Tanto $A \vdash A \vee B$ quanto $B \vdash A \vee B$.

Prova. 1. Considere a seguinte derivação:

$$\dfrac{A \vee B \qquad \dfrac{\neg A \qquad [A]^1}{\bot}\,\neg\text{Elim} \qquad \dfrac{\neg B \qquad [B]^1}{\bot}\,\neg\text{Elim}}{\bot}\,\vee\text{Elim}\ {}^1$$

Esta é uma derivação de $\bot$ a partir das hipóteses não descarregadas $A \vee B$, $\neg A$ e $\neg B$.

2. Podemos derivar ambos:

$$\dfrac{A}{A \vee B}\,\vee\text{Intro} \qquad \dfrac{B}{A \vee B}\,\vee\text{Intro}$$

□

Proposição 11.24. *1.* $A, A \to B \vdash B$.

2. Tanto $\neg A \vdash A \to B$ *quanto* $B \vdash A \to B$.

Prova. 1. Podemos derivar:

$$\dfrac{A \to B \qquad A}{B} \to\text{Elim}$$

2. Isso é mostrado pelas duas derivações seguintes:

$$\dfrac{\dfrac{\neg A \qquad [A]^1}{\dfrac{\bot}{B}\,\bot_I}\,\neg\text{Elim}}{1\ \dfrac{}{A \to B}\,\to\text{Intro}} \qquad\qquad \dfrac{B}{A \to B}\,\to\text{Intro}$$

Note que $\to$Intro pode, mas não precisa, descarregar a hipótese A. □

11.10 Derivabilidade e os Quantificadores

O teorema da completude também requer que as regras de dedução natural produzam os fatos sobre $\vdash$ estabelecidos nesta seção.

Teorema 11.25. *Se* c *é uma constante que não ocorre em* Γ *ou* $A(x)$ *e* $\Gamma \vdash A(c)$*, então* $\Gamma \vdash \forall x\, A(x)$.

Prova. Seja δ uma derivação de $A(c)$ a partir de Γ. Adicionando uma inferência $\forall$Intro, obtemos uma derivação de $\forall x\, A(x)$. Como c não ocorre em Γ ou $A(x)$, a condição de variável própria é satisfeita. □

Proposição 11.26. *1.* $A(t) \vdash \exists x\, A(x)$.

2. $\forall x\, A(x) \vdash A(t)$.

Prova. 1. A seguinte é uma derivação de $\exists x\, A(x)$ a partir de $A(t)$:

$$\dfrac{A(t)}{\exists x\, A(x)}\ \exists\text{Intro}$$

2. A seguinte é uma derivação de $A(t)$ a partir de $\forall x\, A(x)$:

$$\dfrac{\forall x\, A(x)}{A(t)}\ \forall\text{Elim}$$

□

11.11 Correção

Um sistema de derivação, como a dedução natural, é *correto* se não pode derivar coisas que de fato não seguem. A correção é assim uma espécie de propriedade de segurança garantida para sistemas de derivação. Dependendo de qual propriedade da Teoria da Prova está em questão, gostaríamos de saber, por exemplo, que

1. toda sentença derivável é válida;
2. se uma sentença é derivável a partir de outras, ela também é uma consequência delas;
3. se um conjunto de sentenças é inconsistente, ele é insatisfatível.

Estas são propriedades importantes de um sistema de derivação. Se alguma delas não vale, o sistema de derivação é deficiente—ele deriva demais. Consequentemente, estabelecer a correção de um sistema de derivação é da maior importância.

Teorema 11.27 (Correção). *Se A é derivável a partir das hipóteses não descarregadas Γ, então $\Gamma \vDash A$.*

Prova. Seja δ uma derivação de A. Procedemos por indução no número de inferências em δ.

Para a base da indução, mostramos a afirmação quando o número de inferências é 0. Neste caso, δ consiste apenas de uma única sentença A, isto é, uma hipótese. Essa hipótese é não descarregada, pois hipóteses só podem ser descarregadas por inferências, e não há inferências. Assim, qualquer estrutura $\mathfrak{M}$ que satisfaz todas as hipóteses não descarregadas da prova também satisfaz A.

Agora o passo indutivo. Suponha que δ contém n inferências. A(s) premissa(s) da inferência mais baixa são derivadas usando sub-derivações, cada uma das quais contém menos de n inferências. Assumimos a hipótese de indução: as premissas da inferência mais baixa seguem das hipóteses não descarregadas das sub-derivações que terminam nessas premissas. Temos que mostrar que a conclusão A segue das hipóteses não descarregadas de toda a prova.

Distinguimos casos de acordo com o tipo da inferência mais baixa. Primeiro, consideramos as possíveis inferências com apenas uma premissa.

1. Suponha que a última inferência é ¬Intro: a derivação tem a forma

$$\dfrac{\begin{array}{c}\Gamma, [A]^n \\ \vdots\,\delta_1 \\ \bot\end{array}}{\neg A}\, n\ \neg\text{Intro}$$

   Pela hipótese de indução, $\bot$ segue das hipóteses não descarregadas $\Gamma \cup \{A\}$ de δ_1. Considere uma estrutura $\mathfrak{M}$. Precisamos mostrar que, se $\mathfrak{M} \vDash \Gamma$, então $\mathfrak{M} \vDash \neg A$. Suponha por redução ao absurdo que $\mathfrak{M} \vDash \Gamma$, mas $\mathfrak{M} \nvDash \neg A$, isto

é, $M \vDash A$. Isto significaria que $M \vDash \Gamma \cup \{A\}$. Isto é contrário à nossa hipótese de indução. Logo, $M \vDash \neg A$.

2. A última inferência é $\wedge$Elim: há duas variantes: A ou B pode ser inferido da premissa $A \wedge B$. Considere o primeiro caso. A derivação δ tem este aspecto:

$$\dfrac{\begin{array}{c}\Gamma \\ \vdots\,\delta_1 \\ \vdots \\ A \wedge B\end{array}}{A}\ \wedge\text{Elim}$$

Pela hipótese de indução, $A \wedge B$ segue das hipóteses não descarregadas Γ de δ_1. Considere uma estrutura M. Precisamos mostrar que, se $M \vDash \Gamma$, então $M \vDash A$. Suponha $M \vDash \Gamma$. Pela nossa hipótese de indução ($\Gamma \vDash A \wedge B$), sabemos que $M \vDash A \wedge B$. Por definição, $M \vDash A \wedge B$ se, e somente se, $M \vDash A$ e $M \vDash B$. (O caso em que B é inferido de $A \wedge B$ é tratado de forma similar.)

3. A última inferência é $\vee$Intro: há duas variantes: $A \vee B$ pode ser inferido da premissa A ou da premissa B. Considere o primeiro caso. A derivação tem a forma

$$\dfrac{\begin{array}{c}\Gamma \\ \vdots\,\delta_1 \\ \vdots \\ A\end{array}}{A \vee B}\ \vee\text{Intro}$$

Pela hipótese de indução, A segue das hipóteses não descarregadas Γ de δ_1. Considere uma estrutura M. Precisamos mostrar que, se $M \vDash \Gamma$, então $M \vDash A \vee B$. Suponha $M \vDash \Gamma$; então $M \vDash A$, pois $\Gamma \vDash A$ (a hipótese de indução). Logo também deve ser o caso que $M \vDash A \vee B$. (O caso em que $A \vee B$ é inferido de B é tratado de forma similar.)

4. A última inferência é →Intro: $A \to B$ é inferido de uma subprova com hipótese A e conclusão B, isto é,

$$\dfrac{\begin{array}{c}\Gamma, [A]^n \\ \vdots\,\delta_1 \\ B\end{array}}{A \to B}\ {}^{n}\!\to\text{Intro}$$

   Pela hipótese de indução, B segue das hipóteses não descarregadas de δ_1, isto é, $\Gamma \cup \{A\} \models B$. Considere uma estrutura $\mathcal{M}$. As hipóteses não descarregadas de δ são apenas Γ, pois A é descartada na última inferência. Precisamos mostrar que $\Gamma \models A \to B$. Por redução ao absurdo, suponha que para alguma estrutura $\mathcal{M}$, $\mathcal{M} \models \Gamma$ mas $\mathcal{M} \not\models A \to B$. Assim, $\mathcal{M} \models A$ e $\mathcal{M} \not\models B$. Mas por hipótese, B é uma consequência de $\Gamma \cup \{A\}$, isto é, $\mathcal{M} \models B$, o que é uma contradição. Logo, $\Gamma \models A \to B$.

5. A última inferência é $\bot_I$: aqui, δ termina em

$$\dfrac{\begin{array}{c}\Gamma \\ \vdots\,\delta_1 \\ \bot\end{array}}{A}\ \bot_I$$

   Pela hipótese de indução, $\Gamma \models \bot$. Temos que mostrar que $\Gamma \models A$. Suponha o contrário; então para algum $\mathcal{M}$ temos $\mathcal{M} \models \Gamma$ e $\mathcal{M} \not\models A$. Mas sempre temos $\mathcal{M} \not\models \bot$, de modo que isto significaria que $\Gamma \not\models \bot$, contrariando a hipótese de indução.

6. A última inferência é $\bot_C$: Exercício.

7. A última inferência é ∀Intro: então δ tem a forma

$$\dfrac{\begin{array}{c}\Gamma\\ \vdots\\ \delta_1\\ \vdots\\ A(a)\end{array}}{\forall x\, A(x)}\ \forall\text{Intro}$$

A premissa $A(a)$ é consequência das hipóteses não descarregadas Γ pela hipótese de indução. Considere alguma estrutura M tal que $M \vDash \Gamma$. Precisamos mostrar que $M \vDash \forall x\, A(x)$. Como $\forall x\, A(x)$ é uma sentença, isto significa que temos que mostrar que para toda atribuição de variáveis s, $M, s \vDash A(x)$ (Proposição 7.19). Como Γ consiste inteiramente em sentenças, $M, s \vDash B$ para todo $B \in \Gamma$ por Definição 7.11. Seja M' como M, exceto que $a^{M'} = s(x)$. Como a não ocorre em Γ, $M' \vDash \Gamma$ por Corolário 7.21. Como $\Gamma \vDash A(a)$, $M' \vDash A(a)$. Como $A(a)$ é uma sentença, $M', s \vDash A(a)$ por Proposição 7.18. $M', s \vDash A(x)$ se, e somente se, $M' \vDash A(a)$ por Proposição 7.23 (lembre que $A(a)$ é apenas $A(x)[a/x]$). Assim, $M', s \vDash A(x)$. Como a não ocorre em $A(x)$, por Proposição 7.20, $M, s \vDash A(x)$. Mas s foi uma atribuição de variáveis arbitrária, de modo que $M \vDash \forall x\, A(x)$.

8. A última inferência é ∃Intro: Exercício.

9. A última inferência é ∀Elim: Exercício.

Agora consideremos as possíveis inferências com várias premissas: ∨Elim, ∧Intro, →Elim, e ∃Elim.

1. A última inferência é ∧Intro. $A \wedge B$ é inferido das premissas A e B e δ tem a forma

$$\dfrac{\begin{array}{cc}\Gamma_1 & \Gamma_2\\ \vdots & \vdots\\ \delta_1 & \delta_2\\ \vdots & \vdots\\ A & B\end{array}}{A \wedge B}\ \wedge\text{Intro}$$

Pela hipótese de indução, A segue das hipóteses não descarregadas Γ_1 de δ_1 e B segue das hipóteses não descarregadas Γ_2 de δ_2. As hipóteses não descarregadas de δ são $\Gamma_1 \cup \Gamma_2$, de modo que temos que mostrar que $\Gamma_1 \cup \Gamma_2 \vDash A \wedge B$. Considere uma estrutura $\mathfrak{M}$ com $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2$. Como $\mathfrak{M} \vDash \Gamma_1$, deve ser o caso que $\mathfrak{M} \vDash A$, pois $\Gamma_1 \vDash A$, e como $\mathfrak{M} \vDash \Gamma_2$, $\mathfrak{M} \vDash B$, pois $\Gamma_2 \vDash B$. Juntas, $\mathfrak{M} \vDash A \wedge B$.

2. A última inferência é $\vee$Elim: Exercício.

3. A última inferência é $\rightarrow$Elim. B é inferido das premissas $A \rightarrow B$ e A. A derivação δ tem este aspecto:

$$\dfrac{\begin{array}{cc} \Gamma_1 & \Gamma_2 \\ \vdots & \vdots \\ \delta_1 & \delta_2 \\ \vdots & \vdots \\ A \rightarrow B & A \end{array}}{B} \rightarrow\text{Elim}$$

Pela hipótese de indução, $A \rightarrow B$ segue das hipóteses não descarregadas Γ_1 de δ_1 e A segue das hipóteses não descarregadas Γ_2 de δ_2. Considere uma estrutura $\mathfrak{M}$. Precisamos mostrar que, se $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2$, então $\mathfrak{M} \vDash B$. Suponha $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2$. Como $\Gamma_1 \vDash A \rightarrow B$, $\mathfrak{M} \vDash A \rightarrow B$. Como $\Gamma_2 \vDash A$, temos $\mathfrak{M} \vDash A$. Isto significa que $\mathfrak{M} \vDash B$ (pois se $\mathfrak{M} \nvDash B$, como $\mathfrak{M} \vDash A$, teríamos $\mathfrak{M} \nvDash A \rightarrow B$, contradizendo $\mathfrak{M} \vDash A \rightarrow B$).

4. A última inferência é $\neg$Elim: Exercício.

5. A última inferência é $\exists$Elim: Exercício. □

Corolário 11.28. *Se $\vdash A$, então A é válida.*

Corolário 11.29. *Se Γ é satisfatível, então é consistente.*

Prova. Provamos a contrapositiva. Suponha que Γ não é consistente. Então $\Gamma \vdash \bot$, isto é, há uma derivação de $\bot$ a partir de hipóteses não descarregadas em Γ. Por Teorema 11.27, qualquer estrutura M que satisfaz Γ deve satisfazer $\bot$. Como $M \nvDash \bot$ para toda estrutura M, nenhuma M pode satisfazer Γ, isto é, Γ não é satisfatível. □

11.12 Derivações com Identidade

As Derivações com identidade requerem regras de inferência adicionais.

$$\frac{}{t = t}\ =\text{Intro} \qquad \frac{t_1 = t_2 \qquad A(t_1)}{A(t_2)}\ =\text{Elim} \qquad \frac{t_1 = t_2 \qquad A(t_2)}{A(t_1)}\ =\text{Elim}$$

Nas regras acima, t, t_1 e t_2 são termos fechados. A regra =Intro nos permite derivar qualquer afirmação de identidade da forma $t = t$ diretamente, sem hipóteses.

Exemplo 11.30. Se s e t são termos fechados, então $A(s), s = t \vdash A(t)$:

$$\frac{s = t \qquad A(s)}{A(t)}\ =\text{Elim}$$

Isto pode ser familiar como o "princípio da substituibilidade dos identicos", ou Lei de Leibniz.

Exemplo 11.31. Derivamos a sentença

$$\forall x \forall y\, ((A(x) \wedge A(y)) \rightarrow x = y)$$

a partir da sentença

$$\exists x \, \forall y \, (A(y) \to y = x)$$

Desenvolvemos a derivação de trás para frente:

$$\dfrac{\dfrac{\dfrac{\exists x \, \forall y \, (A(y) \to y = x) \qquad [A(a) \wedge A(b)]^1}{\vdots}}{\dfrac{\dfrac{\dfrac{a = b}{((A(a) \wedge A(b)) \to a = b)}\,1 \to\text{Intro}}{\forall y \, ((A(a) \wedge A(y)) \to a = y)} \forall\text{Intro}}{\forall x \, \forall y \, ((A(x) \wedge A(y)) \to x = y)} \forall\text{Intro}}$$

Agora teremos que usar a hipótese principal: como é uma fórmula existencial, usamos ∃Elim para derivar a conclusão intermediária $a = b$.

$$\dfrac{\dfrac{\dfrac{\exists x \, \forall y \, (A(y) \to y = x) \qquad \begin{array}{c}[\forall y \, (A(y) \to y = c)]^2 \\ [A(a) \wedge A(b)]^1 \\ \vdots \\ a = b\end{array}}{a = b}\,2 \exists\text{Elim}}{((A(a) \wedge A(b)) \to a = b)}\,1 \to\text{Intro}}{\forall y \, ((A(a) \wedge A(y)) \to a = y)} \forall\text{Intro}}{\forall x \, \forall y \, ((A(x) \wedge A(y)) \to x = y)} \forall\text{Intro}$$

A sub-derivação no canto superior direito é completada usando suas hipóteses para mostrar que $a = c$ e $b = c$. Isso requer duas derivações separadas. A derivação para $a = c$ é a seguinte:

$$\dfrac{\dfrac{[\forall y \, (A(y) \to y = c)]^2}{A(a) \to a = c} \forall\text{Elim} \qquad \dfrac{[A(a) \wedge A(b)]^1}{A(a)} \wedge\text{Elim}}{a = c} \to\text{Elim}$$

De $a = c$ e $b = c$ derivamos $a = b$ por =Elim.

11.13 Correção com Identidade

Proposição 11.32. *A dedução natural com regras para = é correta.*

Prova. Qualquer fórmula da forma $t = t$ é válida, pois para toda estrutura $\mathfrak{M}$, $\mathfrak{M} \vDash t = t$. (Note que assumimos que o termo t é fechado, isto é, não contém variáveis, de modo que as atribuições de variáveis são irrelevantes).

Suponha que a última inferência em uma derivação é =Elim, isto é, a derivação tem a seguinte forma:

$$\dfrac{\begin{array}{cc}\Gamma_1 & \Gamma_2 \\ \vdots & \vdots \\ \delta_1 & \delta_2 \\ \vdots & \vdots \\ t_1 = t_2 & A(t_1)\end{array}}{A(t_2)}\ =\text{Elim}$$

As premissas $t_1 = t_2$ e $A(t_1)$ são derivadas a partir de hipóteses não descarregadas Γ_1 e Γ_2, respectivamente. Queremos mostrar que $A(t_2)$ segue de $\Gamma_1 \cup \Gamma_2$. Considere uma estrutura $\mathfrak{M}$ com $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2$. Pela hipótese de indução, $\mathfrak{M} \vDash A(t_1)$ e $\mathfrak{M} \vDash t_1 = t_2$. Portanto, $\mathrm{Val}^{\mathfrak{M}}(t_1) = \mathrm{Val}^{\mathfrak{M}}(t_2)$. Seja s qualquer atribuição de variáveis e $m = \mathrm{Val}^{\mathfrak{M}}(t_1) = \mathrm{Val}^{\mathfrak{M}}(t_2)$. Por Proposição 7.23, $\mathfrak{M}, s \vDash A(t_1)$ se, e somente se, $\mathfrak{M}, s[m/x] \vDash A(x)$ se, e somente se, $\mathfrak{M}, s \vDash A(t_2)$. Como $\mathfrak{M} \vDash A(t_1)$, temos $\mathfrak{M} \vDash A(t_2)$. □

Resumo

Sistemas de prova fornecem métodos puramente sintáticos para caracterizar consequência e compatibilidade entre sentenças. A **dedução natural** é um desses sistemas de prova. Uma **derivação** nela consiste em uma árvore de fórmulas. As fórmulas mais ao topo em uma derivação são **hipóteses**. Todas as demais fórmulas, para que a derivação seja correta, devem ser corretamente justificadas por uma de várias **regras de inferência**. Elas

vêm em pares: uma regra de introdução e uma regra de eliminação para cada conectivo e quantificador. Por exemplo, se uma fórmula A é justificada por uma regra $\to$Elim, as fórmulas precedentes (as **premissas**) devem ser $B \to A$ e B (para algum B). Algumas regras de inferência também permitem que hipóteses sejam **descarregadas**. Por exemplo, se $A \to B$ é inferida de B usando $\to$Intro, quaisquer ocorrências de A como hipóteses na derivação que leva à premissa B podem ser descarregadas, e recebe um rótulo que também é registrado na inferência.

Se há uma derivação com fórmula final A e todas as hipóteses estão descarregadas, dizemos que A é um **teorema** e escrevemos $\vdash A$. Se todas as hipóteses não descarregadas estão em algum conjunto Γ, dizemos que A é **derivável de** Γ e escrevemos $\Gamma \vdash A$. Se $\Gamma \vdash \bot$, dizemos que Γ é **inconsistente**; caso contrário, **consistente**. Essas noções são inter-relacionadas; por exemplo, $\Gamma \vdash A$ se e somente se $\Gamma \cup \{\neg A\}$ é inconsistente. Elas são também relacionadas às noções semânticas correspondentes; por exemplo, se $\Gamma \vdash A$, então $\Gamma \vDash A$. Essa propriedade dos sistemas de prova—o que pode ser derivado de Γ tem garantia de ser consequência de Γ—é chamada de **correção**. O **teorema da correção** é provado por indução sobre o comprimento das derivações, mostrando que cada inferência individual preserva a consequência de sua conclusão a partir das hipóteses abertas, desde que suas premissas sejam consequência de suas hipóteses não descarregadas.

Problemas

Problema 11.1. Dê derivações que mostrem o seguinte:

1. $A \wedge (B \wedge C) \vdash (A \wedge B) \wedge C$.
2. $A \vee (B \vee C) \vdash (A \vee B) \vee C$.
3. $A \to (B \to C) \vdash B \to (A \to C)$.
4. $A \vdash \neg\neg A$.

Problema 11.2. Dê derivações que mostrem o seguinte:

1. $(A \vee B) \to C \vdash A \to C$.
2. $(A \to C) \wedge (B \to C) \vdash (A \vee B) \to C$.
3. $\vdash \neg(A \wedge \neg A)$.
4. $B \to A \vdash \neg A \to \neg B$.
5. $\vdash (A \to \neg A) \to \neg A$.
6. $\vdash \neg(A \to B) \to \neg B$.
7. $A \to C \vdash \neg(A \wedge \neg C)$.
8. $A \wedge \neg C \vdash \neg(A \to C)$.
9. $A \vee B, \neg B \vdash A$.
10. $\neg A \vee \neg B \vdash \neg(A \wedge B)$.
11. $\vdash (\neg A \wedge \neg B) \to \neg(A \vee B)$.
12. $\vdash \neg(A \vee B) \to (\neg A \wedge \neg B)$.

Problema 11.3. Dê derivações que mostrem o seguinte:

1. $\neg(A \to B) \vdash A$.
2. $\neg(A \wedge B) \vdash \neg A \vee \neg B$.
3. $A \to B \vdash \neg A \vee B$.
4. $\vdash \neg\neg A \to A$.
5. $A \to B, \neg A \to B \vdash B$.
6. $(A \wedge B) \to C \vdash (A \to C) \vee (B \to C)$.
7. $(A \to B) \to A \vdash A$.
8. $\vdash (A \to B) \vee (B \to C)$.

(Todos exigem a regra $\bot_C$.)

Problema 11.4. Dê derivações que mostrem o seguinte:

1. $\vdash (\forall x\, A(x) \land \forall y\, B(y)) \to \forall z\, (A(z) \land B(z))$.
2. $\vdash (\exists x\, A(x) \lor \exists y\, B(y)) \to \exists z\, (A(z) \lor B(z))$.
3. $\forall x\, (A(x) \to B) \vdash \exists y\, A(y) \to B$.
4. $\forall x\, \neg A(x) \vdash \neg \exists x\, A(x)$.
5. $\vdash \neg \exists x\, A(x) \to \forall x\, \neg A(x)$.
6. $\vdash \neg \exists x\, \forall y\, ((A(x,y) \to \neg A(y,y)) \land (\neg A(y,y) \to A(x,y)))$.

Problema 11.5. Dê derivações que mostrem o seguinte:

1. $\vdash \neg \forall x\, A(x) \to \exists x\, \neg A(x)$.
2. $(\forall x\, A(x) \to B) \vdash \exists y\, (A(y) \to B)$.
3. $\vdash \exists x\, (A(x) \to \forall y\, A(y))$.

(Todas exigem a regra $\bot_C$.)

Problema 11.6. Prove Proposição 11.16

Problema 11.7. Prove que $\Gamma \vdash \neg A$ se, e somente se, $\Gamma \cup \{A\}$ é inconsistente.

Problema 11.8. Complete a prova de Teorema 11.27.

Problema 11.9. Prove que $=$ é simétrica e transitiva, isto é, dê derivações de $\forall x\, \forall y\, (x = y \to y = x)$ e $\forall x\, \forall y\, \forall z ((x = y \land y = z) \to x = z)$

Problema 11.10. Dê derivações das seguintes fórmulas:

1. $\forall x\, \forall y\, ((x = y \land A(x)) \to A(y))$
2. $\exists x\, A(x) \land \forall y\, \forall z\, ((A(y) \land A(z)) \to y = z) \to \exists x\, (A(x) \land \forall y\, (A(y) \to y = x))$

CAPÍTULO 12

O Teorema da Completude

12.1 Introdução

O teorema da completude é um dos resultados mais fundamentais sobre lógica. Ele vem em duas formulações, cuja equivalência iremos provar. Em sua primeira formulação, ele diz algo fundamental sobre a relação entre consequência semântica e nosso sistema de derivação: se uma sentença A segue de algumas sentenças Γ, então também existe uma derivação que estabelece $\Gamma \vdash A$. Assim, o sistema de derivação é tão forte quanto pode possivelmente ser sem provar coisas que de fato não se seguem.

Em sua segunda formulação, ele pode ser enunciado como um resultado de existência de modelo: todo conjunto consistente de sentenças é satisfatível. A consistência é uma noção da teoria da prova: ela diz que nosso sistema de derivação é incapaz de produzir certas derivações. Mas quem garante que, só porque não há derivações de um certo tipo a partir de Γ, está garantido que existe uma estrutura M? Antes de o teorema da completude ser provado pela primeira vez — na verdade, antes de termos os sistemas de derivação que temos hoje — o grande matemático alemão David Hilbert defendia a visão de que a consistência

das teorias matemáticas garante a existência dos objetos de que elas tratam. Ele colocou isso da seguinte forma em uma carta a Gottlob Frege:

> Se os axiomas dados arbitrariamente não se contradizem uns aos outros com todas as suas consequências, então eles são verdadeiros e as coisas definidas pelos axiomas existem. Este é para mim o critério de verdade e existência.

Frege discordou veementemente. A segunda formulação do teorema da completude mostra que Hilbert estava certo pelo menos no sentido de que, se os axiomas são consistentes, então existe *alguma* estrutura que os torna todos verdadeiros.

Estas não são as únicas razões pelas quais o teorema da completude — ou melhor, sua prova — é importante. Ele tem uma série de consequências importantes, algumas das quais discutiremos separadamente. Por exemplo, como toda derivação que mostra $\Gamma \vdash A$ é finita e, portanto, só pode usar uma quantidade finita das sentenças em Γ, segue do teorema da completude que, se A é uma consequência de Γ, então já é uma consequência de um subconjunto finito de Γ. Isso é chamado de *compacidade*. Equivalentemente, se todo subconjunto finito de Γ é consistente, então o próprio Γ deve ser consistente.

Embora o teorema da compacidade decorra do teorema da completude pelo desvio através de derivações, também é possível usar a *prova do* teorema da completude para estabelecê-lo diretamente. Pois o que a prova faz é tomar um conjunto de sentenças com uma certa propriedade — a consistência — e construir uma estrutura a partir desse conjunto que possui certas propriedades (neste caso, que ela satisfaz o conjunto). Quase exatamente a mesma construção pode ser usada para estabelecer a compacidade diretamente, partindo de conjuntos "finitamente satisfatíveis" de sentenças em vez de conjuntos consistentes. A construção também produz outras consequências, por exemplo, que todo conjunto satisfatível de sentenças tem um modelo finito

ou infinitos contáveis. (Esse resultado é chamado de teorema de Löwenheim–Skolem.) Em geral, a construção de estruturas a partir de conjuntos de sentenças é usada com frequência em lógica, e às vezes até em filosofia.

12.2 Esboço da Prova

A prova do teorema da completude é um pouco complexa e, em uma primeira leitura, é fácil se perder. Então, vamos esboçar a prova. O primeiro passo é uma mudança de perspectiva, que nos permite enxergar um caminho para uma prova. Quando a completude é pensada como "sempre que $\Gamma \vDash A$ então $\Gamma \vdash A$", pode ser difícil até mesmo ter uma ideia: pois, para mostrar que $\Gamma \vdash A$, temos que encontrar uma derivação, e não parece que a hipótese de que $\Gamma \vDash A$ nos ajude nisso de alguma forma. Para alguns sistemas de prova, é possível construir diretamente uma derivação, mas adotaremos uma abordagem um pouco diferente. A mudança de perspectiva necessária é esta: a completude também pode ser formulada como: "se Γ é consistente, então é satisfatível". Talvez possamos usar a informação em Γ juntamente com a hipótese de que ela é consistente para construir uma estrutura que satisfaça toda sentença em Γ. Afinal, sabemos que tipo de estrutura estamos procurando: uma que seja como Γ a descreve!

Se Γ contém apenas sentenças atômicas, é fácil construir um modelo para ele. Suponha que as sentenças atômicas sejam todas da forma $P(a_1, \ldots, a_n)$ onde os a_i são símbolos de constante. Tudo o que temos a fazer é apresentar um domínio $|\boldsymbol{M}|$ e uma atribuição para P de modo que $\boldsymbol{M} \vDash P(a_1, \ldots, a_n)$. Mas isso não é muito difícil: faça $|\boldsymbol{M}| = \mathbb{N}$, $c_i^{\boldsymbol{M}} = i$ e, para cada $P(a_1, \ldots, a_n) \in \Gamma$, coloque a tupla $\langle k_1, \ldots, k_n \rangle$ em $P^{\boldsymbol{M}}$, onde k_i é o índice do símbolo de constante a_i (isto é, $a_i \equiv c_{k_i}$).

Agora suponha que Γ contenha alguma fórmula $\neg B$, com B atômica. Poderíamos temer que a construção de $\boldsymbol{M}$ interfira na possibilidade de tornar $\neg B$ verdadeira. Mas é aqui que entra a consistência de Γ: se $\neg B \in \Gamma$, então $B \notin \Gamma$, caso contrário Γ

seria inconsistente. E se $B \notin \Gamma$, então, de acordo com nossa construção de M, $M \not\vDash B$, logo $M \vDash \neg B$. Até aqui, tudo bem.

E se Γ contiver fórmulas complexas, não atômicas? Digamos que ele contenha $A \wedge B$. Para tornar isso verdadeiro, devemos proceder como se tanto A quanto B estivessem em Γ. E se $A \vee B \in \Gamma$, então teremos que tornar pelo menos um deles verdadeiro, isto é, proceder como se um deles estivesse em Γ.

Isso sugere a seguinte ideia: adicionamos fórmulas adicionais a Γ de modo a (a) manter o conjunto resultante consistente e (b) garantir que, para toda sentença A atômica possível, ou A está no conjunto resultante, ou $\neg A$ está, e (c) tais que, sempre que $A \wedge B$ está no conjunto, então tanto A quanto B também estão; se $A \vee B$ está no conjunto, pelo menos um dentre A ou B também está, etc. Continuamos fazendo isso (potencialmente para sempre). Chame o conjunto de todas as fórmulas assim adicionadas de Γ^*. Então a nossa construção acima nos forneceria uma estrutura M para a qual poderíamos provar, por indução, que ela satisfaz todas as sentenças em Γ^* e, portanto, também todas as sentenças em Γ, já que $\Gamma \subseteq \Gamma^*$. Acontece que garantir (a) e (b) é suficiente. Um conjunto de sentenças para o qual (b) vale é chamado de *completo.* Assim, nossa tarefa será estender o conjunto consistente Γ a um conjunto consistente e completo Γ^*.

Há um problema neste plano: se $\exists x\, A(x) \in \Gamma$, esperaríamos poder escolher algum símbolo de constante c e adicionar $A(c)$ neste processo. Mas como sabemos que sempre podemos fazer isso? Talvez tenhamos apenas alguns símbolos de constante em nossa linguagem, e, para cada um deles, tenhamos $\neg A(c) \in \Gamma$. Também não podemos adicionar $A(c)$, já que isso tornaria o conjunto inconsistente, e não saberíamos se M tem que tornar $A(c)$ ou $\neg A(c)$ verdadeira. Além disso, pode acontecer de Γ conter apenas sentenças em uma linguagem que não tem símbolo de constante algum (por exemplo, a linguagem da teoria dos conjuntos).

A solução para este problema é simplesmente adicionar infinitas constantes no começo, além de sentenças que as conectam

com os quantificadores da maneira correta. (Claro, temos que verificar que isso não pode introduzir uma inconsistência.)

Nossa construção original funciona bem se tivermos apenas símbolos de constante nas sentenças atômicas. Mas a linguagem também pode conter símbolos de função. Nesse caso, pode ser complicado encontrar as funções certas em $\mathbb{N}$ para atribuir a esses símbolos de função de modo a fazer tudo funcionar. Então, aqui vai outro truque: em vez de usar i para interpretar c_i, basta tomar o próprio conjunto de símbolos de constante como o domínio. Então $\boldsymbol{M}$ pode atribuir cada símbolo de constante a si mesmo: $c_i^{\boldsymbol{M}} = c_i$. Mas por que não ir até o fim: seja $|\boldsymbol{M}|$ o conjunto de todos os *termos* da linguagem! Se fizermos isso, há uma atribuição óbvia de funções (que tomam termos como argumentos e têm termos como valores) aos símbolos de função: atribuímos ao símbolo de função f_i^n a função que, dados n termos $t_1, \ldots, t_n$ como entrada, produz o termo $f_i^n(t_1, \ldots, t_n)$ como valor.

A última peça do quebra-cabeça é o que fazer com =. O predicado = tem uma interpretação fixa: $\boldsymbol{M} \vDash t = t'$ se e somente se $\mathrm{Val}^{\boldsymbol{M}}(t) = \mathrm{Val}^{\boldsymbol{M}}(t')$. Agora, se ajustarmos as coisas de modo que o valor de um termo t seja o próprio t, então esta estrutura não tornará verdadeira *nenhuma* sentença da forma $t = t'$ a menos que t e t' sejam um único e mesmo termo. E, claro, isso é um problema, já que basicamente toda teoria interessante em uma linguagem com símbolos de função terá como teoremas sentenças $t = t'$ onde t e t' não são o mesmo termo (por exemplo, em teorias da aritmética: $(0 + 0) = 0$). Para resolver este problema, mudamos o domínio de $\boldsymbol{M}$: em vez de usar termos como os objetos em $|\boldsymbol{M}|$, usamos conjuntos de termos, e cada conjunto é tal que contém todos aqueles termos que as sentenças em Γ exigem que sejam iguais. Assim, por exemplo, se Γ é uma teoria da aritmética, um desses conjuntos conterá: 0, $(0 + 0)$, (0×0), etc. Este será o conjunto que atribuímos a 0, e acontecerá que este conjunto também é o valor de todos os termos nele, por exemplo, também de $(0 + 0)$. Portanto, a sentença $(0 + 0) = 0$ será verdadeira nesta estrutura revisada.

Então, aqui está o que faremos. Primeiro investigamos as propriedades dos conjuntos consistentes completos; em particular, provamos que um conjunto consistente completo contém $A \wedge B$ se e somente se contém tanto A quanto B, contém $A \vee B$ se e somente se contém pelo menos um deles, etc. (Proposição 12.2). Em seguida, definimos e investigamos conjuntos "saturados" de sentenças. Um conjunto saturado é aquele que contém condicionais que ligam cada sentença quantificada às suas instâncias (Definição 12.5). Mostramos que qualquer conjunto consistente Γ sempre pode ser estendido a um conjunto saturado Γ' (Lema 12.6). Se um conjunto é consistente, saturado e completo, ele também tem a propriedade de conter $\exists x\, A(x)$ se e somente se contém $A(t)$ para algum termo fechado t e $\forall x\, A(x)$ se e somente se contém $A(t)$ para todo termo fechado t (Proposição 12.7). Em seguida, tomaremos o conjunto consistente saturado Γ' e mostraremos que ele pode ser estendido a um conjunto saturado, consistente e completo Γ^* (Lema 12.8). Este conjunto Γ^* é o que usaremos para definir nosso modelo de termos $M(\Gamma^*)$. O modelo de termos tem o conjunto de termos fechados como seu domínio, e a interpretação de seus predicados é dada pelas sentenças atômicas em Γ^* (Definição 12.9). Usaremos as propriedades dos conjuntos consistentes completos e saturados para mostrar que, de fato, $M(\Gamma^*) \vDash A$ se e somente se $A \in \Gamma^*$ (Lema 12.12) e, portanto, em particular, $M(\Gamma^*) \vDash \Gamma$. Por fim, consideraremos como definir um modelo de termos se Γ contém $=$ também (Definição 12.16) e mostraremos que ele satisfaz Γ^* (Lema 12.19).

12.3 Conjuntos Consistentes Completos de Sentenças

Definição 12.1 (Conjunto completo). Um conjunto Γ de sentenças é *completo* se e somente se, para qualquer sentença A, ou $A \in \Gamma$ ou $\neg A \in \Gamma$.

Conjuntos completos de sentenças não deixam questões sem resposta. Para qualquer sentença A, Γ "diz" se A é verdadeira ou falsa. A importância dos conjuntos completos estende-se para além da prova do teorema da completude. Uma teoria que é completa e axiomatizável, por exemplo, é sempre decidível.

Conjuntos consistentes completos são importantes na prova de completude já que podemos garantir que todo conjunto consistente de sentenças Γ está contido em um conjunto consistente completo Γ^*. Um conjunto consistente completo contém, para cada sentença A, ou A ou sua negação $\neg A$, mas não ambas. Isso é verdade em particular para sentenças atômicas, de modo que, a partir de um conjunto consistente completo em uma linguagem adequadamente expandida por símbolos de constante, podemos construir uma estrutura onde a interpretação dos predicados é definida de acordo com quais sentenças atômicas estão em Γ^*. Pode-se então mostrar que esta estrutura torna verdadeiras todas as sentenças em Γ^* (e, portanto, também todas aquelas em Γ). A prova deste último fato requer que $\neg A \in \Gamma^*$ se e somente se $A \notin \Gamma^*$, $(A \vee B) \in \Gamma^*$ se e somente se $A \in \Gamma^*$ ou $B \in \Gamma^*$, etc.

No que se segue, usaremos com frequência, de modo tácito, as propriedades de reflexividade, monotonicidade e transitividade de $\vdash$ (ver seções 10.8 e 11.7).

Proposição 12.2. *Suponha que Γ seja completo e consistente. Então:*

1. *Se $\Gamma \vdash A$, então $A \in \Gamma$.*
2. *$A \wedge B \in \Gamma$ se e somente se $A \in \Gamma$ e $B \in \Gamma$.*
3. *$A \vee B \in \Gamma$ se e somente se $A \in \Gamma$ ou $B \in \Gamma$.*
4. *$A \to B \in \Gamma$ se e somente se $A \notin \Gamma$ ou $B \in \Gamma$.*

Prova. Suponhamos, para tudo o que se segue, que Γ seja completo e consistente.

1. Se $\Gamma \vdash A$, então $A \in \Gamma$.

   Suponha que $\Gamma \vdash A$. Suponha, ao contrário, que $A \notin \Gamma$. Como Γ é completo, $\neg A \in \Gamma$. Por Proposições 10.20 e 11.20, Γ é inconsistente. Isso contradiz a suposição de que Γ é consistente. Logo, não pode ser o caso que $A \notin \Gamma$, de modo que $A \in \Gamma$.

2. Exercício.

3. Primeiro mostramos que, se $A \vee B \in \Gamma$, então $A \in \Gamma$ ou $B \in \Gamma$. Suponha $A \vee B \in \Gamma$ mas $A \notin \Gamma$ e $B \notin \Gamma$. Como Γ é completo, $\neg A \in \Gamma$ e $\neg B \in \Gamma$. Por Proposições 10.23 e 11.23, item (1), Γ é inconsistente, uma contradição. Logo, $A \in \Gamma$ ou $B \in \Gamma$.

   Para a direção reversa, suponha que $A \in \Gamma$ ou $B \in \Gamma$. Por Proposições 10.23 e 11.23, item (2), $\Gamma \vdash A \vee B$. Por (1), $A \vee B \in \Gamma$, como requerido.

4. Exercício. □

12.4 Expansão de Henkin

Parte do desafio em provar o teorema da completude é que o modelo que construímos a partir de um conjunto consistente completo Γ deve tornar verdadeiras todas as fórmulas quantificadas em Γ. Para garantir isso, usamos um truque devido a Leon Henkin. Em essência, o truque consiste em expandir a linguagem por infinitos símbolos de constante e adicionar, para cada fórmula com uma variável livre $A(x)$, uma fórmula da forma $\exists x\, A(x) \to A(c)$, onde c é um dos novos símbolos de constante. Quando construímos a estrutura que satisfaz Γ, isso garantirá que cada sentença existencial verdadeira tenha uma testemunha entre as novas constantes.

Proposição 12.3. *Se Γ é consistente em $\mathscr{L}$ e $\mathscr{L}'$ é obtida de $\mathscr{L}$ adicionando-se a infinito contável conjunto de novos símbolos de constante d_0, d_1, ..., então Γ é consistente em $\mathscr{L}'$.*

Definição 12.4 (Conjunto saturado). Um conjunto Γ de fórmulas de uma linguagem $\mathscr{L}$ é *saturado* se e somente se, para cada fórmula $A(x) \in \mathrm{Frm}(\mathscr{L})$ com uma variável livre x, existe um símbolo de constante $c \in \mathscr{L}$ tal que $\exists x\, A(x) \to A(c) \in \Gamma$.

A definição a seguir será usada na prova do próximo teorema.

Definição 12.5. Seja $\mathscr{L}'$ como em Proposição 12.3. Fixe uma enumeração $A_0(x_0)$, $A_1(x_1)$, ... de todas as fórmulas $A_i(x_i)$ de $\mathscr{L}'$ nas quais uma variável (x_i) ocorre livre. Definimos as sentenças D_n por indução sobre n.

Seja c_0 o primeiro símbolo de constante dentre os d_i que adicionamos a $\mathscr{L}$ que não ocorre em $A_0(x_0)$. Supondo que D_0, ..., D_{n-1} já tenham sido definidas, seja c_n o primeiro dentre os novos símbolos de constante d_i que não ocorre nem em D_0, ..., D_{n-1} nem em $A_n(x_n)$.

Agora seja D_n a fórmula $\exists x_n\, A_n(x_n) \to A_n(c_n)$.

Lema 12.6. *Todo conjunto consistente Γ pode ser estendido a um conjunto saturado consistente Γ'.*

Prova. Dado um conjunto consistente de sentenças Γ em uma linguagem $\mathscr{L}$, expanda a linguagem adicionando a infinito contável conjunto de novos símbolos de constante para formar $\mathscr{L}'$. Por Proposição 12.3, Γ ainda é consistente na linguagem mais rica. Além disso, seja D_i como em Definição 12.5. Seja

$$\Gamma_0 = \Gamma$$
$$\Gamma_{n+1} = \Gamma_n \cup \{D_n\}$$

isto é, $\Gamma_{n+1} = \Gamma \cup \{D_0, \ldots, D_n\}$, e seja $\Gamma' = \bigcup_n \Gamma_n$. Γ' é claramente saturado.

Se Γ' fosse inconsistente, então, para algum n, Γ_n seria inconsistente (Exercício: explique por quê). Portanto, para mostrar que Γ' é consistente, basta mostrar, por indução sobre n, que cada conjunto Γ_n é consistente.

A base da indução é simplesmente a afirmação de que $\Gamma_0 = \Gamma$ é consistente, que é a hipótese do teorema. Para o passo de indução, suponha que Γ_n seja consistente mas $\Gamma_{n+1} = \Gamma_n \cup \{D_n\}$ seja inconsistente. Lembre que D_n é $\exists x_n\, A_n(x_n) \to A_n(c_n)$, onde $A_n(x_n)$ é uma fórmula de $\mathscr{L}'$ com apenas a variável x_n livre. Pela maneira como escolhemos os c_n (ver Definição 12.5), c_n não ocorre em $A_n(x_n)$ nem em Γ_n.

Se $\Gamma_n \cup \{D_n\}$ é inconsistente, então $\Gamma_n \vdash \neg D_n$ e, portanto, ambos os seguintes valem:

$$\Gamma_n \vdash \exists x_n\, A_n(x_n) \qquad \Gamma_n \vdash \neg A_n(c_n)$$

Como c_n não ocorre em Γ_n ou em $A_n(x_n)$, Teoremas 10.25 e 11.25 se aplica. De $\Gamma_n \vdash \neg A_n(c_n)$, obtemos $\Gamma_n \vdash \forall x_n \neg A_n(x_n)$. Assim, temos que ambos $\Gamma_n \vdash \exists x_n\, A_n(x_n)$ e $\Gamma_n \vdash \forall x_n \neg A_n(x_n)$, de modo que o próprio Γ_n é inconsistente. (Note que $\forall x_n \neg A_n(x_n) \vdash \neg\exists x_n\, A_n(x_n)$.) Contradição: supunha-se que Γ_n fosse consistente. Logo, $\Gamma_n \cup \{D_n\}$ é consistente. □

Mostraremos agora que conjuntos *completos*, consistentes que são saturados têm a propriedade de que contêm uma sentença universalmente quantificada se e somente se contêm todas as suas instâncias e contêm uma sentença existencialmente quantificada se e somente se contêm pelo menos uma instância. Usaremos isso para mostrar que a estrutura que geraremos a partir de um conjunto completo, consistente, saturado torna todas as suas sentenças quantificadas verdadeiras.

Proposição 12.7. *Suponha que Γ seja completo, consistente e saturado.*

1. *$\exists x\, A(x) \in \Gamma$ se e somente se $A(t) \in \Gamma$ para pelo menos um termo fechado t.*

2. *$\forall x\, A(x) \in \Gamma$ se e somente se $A(t) \in \Gamma$ para todo termo fechado t.*

Prova. 1. Suponha primeiro que $\exists x\, A(x) \in \Gamma$. Como Γ é saturado, $(\exists x\, A(x) \to A(c)) \in \Gamma$ para algum símbolo de constante c. Por Proposições 10.24 e 11.24, item (1), e Proposição 12.2(1), $A(c) \in \Gamma$.

Para a outra direção, a saturação não é necessária: Suponha $A(t) \in \Gamma$. Então $\Gamma \vdash \exists x\, A(x)$ por Proposições 10.26 e 11.26, item (1). Por Proposição 12.2(1), $\exists x\, A(x) \in \Gamma$.

2. Exercício. □

12.5 Lema de Lindenbaum

Provamos agora um lema que mostra que qualquer conjunto consistente de sentenças está contido em algum conjunto de sentenças que não é apenas consistente, mas também completo. A prova funciona adicionando uma sentença de cada vez, garantindo a cada passo que o conjunto permanece consistente. Fazemos isso de modo que, para todo A, ou A ou $\neg A$ seja adicionada em algum estágio. A união de todos os estágios dessa construção então contém ou A ou sua negação $\neg A$ e é, portanto, completa. Ela também é consistente, já que nos asseguramos, a cada estágio, de não introduzir uma inconsistência.

Lema 12.8 (Lema de Lindenbaum). *Todo conjunto consistente Γ em uma linguagem $\mathcal{L}$ pode ser estendido a um conjunto consistente completo Γ^*.*

Prova. Seja Γ consistente. Sejam $A_0, A_1, \ldots$ uma enumeração de todas as sentenças de $\mathscr{L}$. Defina $\Gamma_0 = \Gamma$, e

$$\Gamma_{n+1} = \begin{cases} \Gamma_n \cup \{A_n\} & \text{se } \Gamma_n \cup \{A_n\} \text{ é consistente;} \\ \Gamma_n \cup \{\neg A_n\} & \text{caso contrário.} \end{cases}$$

Seja $\Gamma^* = \bigcup_{n \geq 0} \Gamma_n$.

Cada Γ_n é consistente: Γ_0 é consistente por definição. Se $\Gamma_{n+1} = \Gamma_n \cup \{A_n\}$, isso é porque este último é consistente. Se não é, $\Gamma_{n+1} = \Gamma_n \cup \{\neg A_n\}$. Temos que verificar que $\Gamma_n \cup \{\neg A_n\}$ é consistente. Suponha que não seja. Então *ambos* $\Gamma_n \cup \{A_n\}$ e $\Gamma_n \cup \{\neg A_n\}$ são inconsistentes. Isso significa que Γ_n seria inconsistente por Proposições 10.21 e 11.21, contrariando a hipótese de indução.

Para todo n e todo $i < n$, $\Gamma_i \subseteq \Gamma_n$. Isso segue por uma simples indução sobre n. Para $n = 0$, não há $i < 0$, de modo que a afirmação vale automaticamente. Para o passo indutivo, suponha que ela seja verdadeira para n. Mostramos que, se $i < n + 1$, então $\Gamma_i \subseteq \Gamma_{n+1}$. Temos $\Gamma_{n+1} = \Gamma_n \cup \{A_n\}$ ou $= \Gamma_n \cup \{\neg A_n\}$ por construção. Logo $\Gamma_n \subseteq \Gamma_{n+1}$. Se $i < n + 1$, então $\Gamma_i \subseteq \Gamma_n$ por hipótese indutiva (se $i < n$) ou pelo fato trivial de que $\Gamma_n \subseteq \Gamma_n$ (se $i = n$). Obtemos que $\Gamma_i \subseteq \Gamma_{n+1}$ por transitividade de $\subseteq$.

Disso segue que Γ^* é consistente. Eis o porquê: Seja $\Gamma' \subseteq \Gamma^*$ finito. Cada $B \in \Gamma'$ também está em Γ_i para algum i. Seja n o maior destes. Como $\Gamma_i \subseteq \Gamma_n$ se $i \leq n$, todo $B \in \Gamma'$ também está $\in \Gamma_n$, isto é, $\Gamma' \subseteq \Gamma_n$, e Γ_n é consistente. Assim, todo subconjunto finito $\Gamma' \subseteq \Gamma^*$ é consistente. Por Proposições 10.17 e 11.17, Γ^* é consistente.

Toda sentença de $\mathrm{Frm}(\mathscr{L})$ aparece na lista usada para definir Γ^*. Se $A_n \notin \Gamma^*$, então isso é porque $\Gamma_n \cup \{A_n\}$ era inconsistente. Mas então $\neg A_n \in \Gamma^*$, de modo que Γ^* é completo. □

12.6 Construção de um Modelo

Por ora, não nos preocupamos com $=$, isto é, queremos apenas mostrar que um conjunto consistente Γ de sentenças que não con-

tém = é satisfatível. Primeiro estendemos Γ a um conjunto consistente, completo e saturado Γ^*. Nesse caso, a definição de um modelo $M(\Gamma^*)$ é simples: Tomamos o conjunto de termos fechados de $\mathscr{L}'$ como o domínio. Atribuímos cada símbolo de constante a si mesmo e nos asseguramos de que, mais geralmente, para todo termo fechado t, $\mathrm{Val}^{M(\Gamma^*)}(t) = t$. Aos predicados são atribuídas extensões de tal forma que uma sentença atômica é verdadeira em $M(\Gamma^*)$ se e somente se está em Γ^*. Isso obviamente tornará verdadeiras em $M(\Gamma^*)$ todas as sentenças atômicas em Γ^*. As demais são verdadeiras desde que o Γ^* com que começamos seja consistente, completo e saturado.

Definição 12.9 (Modelo de termos). Seja Γ^* um conjunto consistente, completo e saturado de sentenças em uma linguagem $\mathscr{L}$. O *modelo de termos* $M(\Gamma^*)$ de Γ^* é a estrutura definida como segue:

1. O domínio $|M(\Gamma^*)|$ é o conjunto de todos os termos fechados de $\mathscr{L}$.

2. A interpretação de um símbolo de constante c é o próprio c: $c^{M(\Gamma^*)} = c$.

3. Ao símbolo de função f é atribuída a função que, dados como argumentos os termos fechados $t_1, \ldots, t_n$, tem como valor o termo fechado $f(t_1, \ldots, t_n)$:

$$f^{M(\Gamma^*)}(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$$

4. Se R é um predicado de n lugares, então

$$\langle t_1, \ldots, t_n \rangle \in R^{M(\Gamma^*)} \text{ se e somente se } R(t_1, \ldots, t_n) \in \Gamma^*.$$

Verificaremos agora que de fato temos $\mathrm{Val}^{M(\Gamma^*)}(t) = t$.

Lema 12.10. *Seja $\mathbf{M}(\Gamma^*)$ o modelo de termos de Definição 12.9; então $\mathrm{Val}^{\mathbf{M}(\Gamma^*)}(t) = t$.*

Prova. A prova é por indução sobre t, onde o caso base, quando t é um símbolo de constante, segue diretamente da definição do modelo de termos. Para o passo de indução, assuma que $t_1, \ldots, t_n$ são termos fechados tais que $\mathrm{Val}^{\mathbf{M}(\Gamma^*)}(t_i) = t_i$ e que f é um símbolo de função n-ária. Então

$$\begin{aligned}\mathrm{Val}^{\mathbf{M}(\Gamma^*)}(f(t_1, \ldots, t_n)) &= f^{\mathbf{M}(\Gamma^*)}(\mathrm{Val}^{\mathbf{M}(\Gamma^*)}(t_1), \ldots, \mathrm{Val}^{\mathbf{M}(\Gamma^*)}(t_n)) \\ &= f^{\mathbf{M}(\Gamma^*)}(t_1, \ldots, t_n) \\ &= f(t_1, \ldots, t_n),\end{aligned}$$

e, assim, por indução, isto vale para todo termo fechado t. □

Uma estrutura $\mathbf{M}$ pode tornar verdadeira uma sentença $\exists x\, A(x)$ existencialmente quantificada sem que haja uma instância $A(t)$ que ela torne verdadeira. Uma estrutura $\mathbf{M}$ pode tornar verdadeiras todas as instâncias $A(t)$ de uma sentença $\forall x\, A(x)$ universalmente quantificada sem tornar $\forall x\, A(x)$ verdadeira. Isso ocorre porque, em geral, nem todo elemento de $|\mathbf{M}|$ é o valor de um termo fechado ($\mathbf{M}$ pode não ser coberta). Esta é a razão pela qual a relação de satisfação é definida por meio de atribuições de variáveis. Entretanto, para o nosso modelo de termos $\mathbf{M}(\Gamma^*)$ isso não seria necessário — porque ela é coberta. Este é o conteúdo do próximo resultado.

Proposição 12.11. *Seja $\mathbf{M}(\Gamma^*)$ o modelo de termos de Definição 12.9.*

1. *$\mathbf{M}(\Gamma^*) \vDash \exists x\, A(x)$ se e somente se $\mathbf{M}(\Gamma^*) \vDash A(t)$ para pelo menos um termo fechado t.*
2. *$\mathbf{M}(\Gamma^*) \vDash \forall x\, A(x)$ se e somente se $\mathbf{M}(\Gamma^*) \vDash A(t)$ para todo termo fechado t.*

Prova. 1. Por Proposição 7.19, $M(\Gamma^*) \vDash \exists x\, A(x)$ se e somente se, para pelo menos uma atribuição de variáveis s, $M(\Gamma^*), s \vDash A(x)$. Como $|M(\Gamma^*)|$ consiste nos termos fechados de $\mathscr{L}$, este é o caso se e somente se há pelo menos um termo fechado t tal que $s(x) = t$ e $M(\Gamma^*), s \vDash A(x)$. Por Proposição 7.23, $M(\Gamma^*), s \vDash A(x)$ se e somente se $M(\Gamma^*), s \vDash A(t)$, onde $s(x) = t$. Por Proposição 7.18, $M(\Gamma^*), s \vDash A(t)$ se e somente se $M(\Gamma^*) \vDash A(t)$, já que $A(t)$ é uma sentença.

2. Exercício. □

Lema 12.12 (Lema da Verdade). *Suponha que A não contenha $=$. Então $M(\Gamma^*) \vDash A$ se e somente se $A \in \Gamma^*$.*

Prova. Provamos ambas as direções simultaneamente e por indução sobre A.

1. $A \equiv \bot$: $M(\Gamma^*) \nvDash \bot$ por definição de satisfação. Por outro lado, $\bot \notin \Gamma^*$ já que Γ^* é consistente.

2. $A \equiv R(t_1, \ldots, t_n)$: $M(\Gamma^*) \vDash R(t_1, \ldots, t_n)$ se e somente se $\langle t_1, \ldots, t_n \rangle \in R^{M(\Gamma^*)}$ (pela definição de satisfação) se e somente se $R(t_1, \ldots, t_n) \in \Gamma^*$ (pela construção de $M(\Gamma^*)$).

3. $A \equiv \neg B$: $M(\Gamma^*) \vDash A$ se e somente se $M(\Gamma^*) \nvDash B$ (por definição de satisfação). Por hipótese de indução, $M(\Gamma^*) \nvDash B$ se e somente se $B \notin \Gamma^*$. Como Γ^* é consistente e completo, $B \notin \Gamma^*$ se e somente se $\neg B \in \Gamma^*$.

4. $A \equiv B \wedge C$: Exercício.

5. $A \equiv B \vee C$: $M(\Gamma^*) \vDash A$ se e somente se $M(\Gamma^*) \vDash B$ ou $M(\Gamma^*) \vDash C$ (por definição de satisfação) se e somente se $B \in \Gamma^*$ ou $C \in \Gamma^*$ (por hipótese de indução). Este é o caso se e somente se $(B \vee C) \in \Gamma^*$ (por Proposição 12.2(3)).

6. $A \equiv B \to C$: Exercício.

7. $A \equiv \forall x\, B(x)$: Exercício.

8. $A \equiv \exists x\, B(x)$: $M(\Gamma^*) \vDash A$ se e somente se $M(\Gamma^*) \vDash B(t)$ para pelo menos um termo t (Proposição 12.11). Por hipótese de indução, este é o caso se e somente se $B(t) \in \Gamma^*$ para pelo menos um termo t. Por Proposição 12.7, isto, por sua vez, é o caso se e somente se $\exists x\, B(x) \in \Gamma^*$. □

12.7 Identidade

A construção do modelo de termos dada na seção anterior é suficiente para estabelecer a completude da lógica de primeira ordem para conjuntos Γ que não contêm $=$. O modelo de termos satisfaz todo $A \in \Gamma^*$ que não contém $=$ (e, portanto, todo $A \in \Gamma$). Ele não funciona, no entanto, se $=$ está presente. A razão é que Γ^* pode então conter uma sentença $t = t'$, mas no modelo de termos o valor de qualquer termo é esse próprio termo. Logo, se t e t' são termos diferentes, seus valores no modelo de termos — isto é, t e t', respectivamente — são diferentes, e assim $t = t'$ é falsa. Podemos corrigir isso, no entanto, usando uma construção conhecida como "fatoração".

Definição 12.13. Seja Γ^* um conjunto consistente e completo de sentenças em $\mathcal{L}$. Definimos a relação $\approx$ sobre o conjunto de termos fechados de $\mathcal{L}$ por

$$t \approx t' \quad \text{se e somente se} \quad t = t' \in \Gamma^*$$

Proposição 12.14. *A relação $\approx$ tem as seguintes propriedades:*

1. *$\approx$ é reflexiva.*

2. *$\approx$ é simétrica.*

3. *$\approx$ é transitiva.*

4. *Se $t \approx t'$, f é um símbolo de função, e $t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n$ são termos fechados, então*

$$f(t_1, \ldots, t_{i-1}, t, t_{i+1}, \ldots, t_n) \approx f(t_1, \ldots, t_{i-1}, t', t_{i+1}, \ldots, t_n).$$

5. *Se $t \approx t'$, R é um predicado, e $t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n$ são termos fechados, então*

$$R(t_1, \ldots, t_{i-1}, t, t_{i+1}, \ldots, t_n) \in \Gamma^* \text{ se e somente se } R(t_1, \ldots, t_{i-1}, t', t_{i+1}, \ldots, t_n) \in \Gamma^*.$$

Prova. Como Γ^* é consistente e completo, $t = t' \in \Gamma^*$ se e somente se $\Gamma^* \vdash t = t'$. Assim, basta mostrar o seguinte:

1. $\Gamma^* \vdash t = t$ para todos os termos fechados t.
2. Se $\Gamma^* \vdash t = t'$ então $\Gamma^* \vdash t' = t$.
3. Se $\Gamma^* \vdash t = t'$ e $\Gamma^* \vdash t' = t''$, então $\Gamma^* \vdash t = t''$.
4. Se $\Gamma^* \vdash t = t'$, então

   $$\Gamma^* \vdash f(t_1, \ldots, t_{i-1}, t, t_{i+1}, , \ldots, t_n) = f(t_1, \ldots, t_{i-1}, t', t_{i+1}, \ldots, t_n)$$

   para todo símbolo de função f de n lugares e termos fechados $t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n$.
5. Se $\Gamma^* \vdash t = t'$ e $\Gamma^* \vdash R(t_1, \ldots, t_{i-1}, t, t_{i+1}, \ldots, t_n)$, então $\Gamma^* \vdash R(t_1, \ldots, t_{i-1}, t', t_{i+1}, \ldots, t_n)$ para todo predicado R de n lugares e termos fechados $t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n$. □

Definição 12.15. Suponha que Γ^* seja um conjunto consistente e completo em uma linguagem $\mathcal{L}$, t seja um termo fechado e $\approx$ como na definição anterior. Então:

$$[t]_\approx = \{t' : t' \in \mathrm{Trm}(\mathcal{L}), t \approx t'\}$$

e $\mathrm{Trm}(\mathscr{L})/_{\approx} = \{[t]_{\approx} : t \in \mathrm{Trm}(\mathscr{L})\}$.

Definição 12.16. Seja $M = M(\Gamma^*)$ o modelo de termos para Γ^* de Definição 12.9. Então $M/_{\approx}$ é a seguinte estrutura:

1. $|M/_{\approx}| = \mathrm{Trm}(\mathscr{L})/_{\approx}$.
2. $c^{M/_{\approx}} = [c]_{\approx}$
3. $f^{M/_{\approx}}([t_1]_{\approx}, \ldots, [t_n]_{\approx}) = [f(t_1, \ldots, t_n)]_{\approx}$
4. $\langle [t_1]_{\approx}, \ldots, [t_n]_{\approx} \rangle \in R^{M/_{\approx}}$ se e somente se $M \vDash R(t_1, \ldots, t_n)$, isto é, se e somente se $R(t_1, \ldots, t_n) \in \Gamma^*$.

Note que definimos $f^{M/_{\approx}}$ e $R^{M/_{\approx}}$ para elementos de $\mathrm{Trm}(\mathscr{L})/_{\approx}$ referindo-nos a eles como $[t]_{\approx}$, isto é, por meio de *representantes* $t \in [t]_{\approx}$. Temos que nos assegurar de que essas definições não dependam da escolha desses representantes, isto é, de que, para algumas outras escolhas t' que determinam as mesmas classes de equivalência ($[t]_{\approx} = [t']_{\approx}$), as definições produzam o mesmo resultado. Por exemplo, se R é um predicado de um lugar, a última cláusula da definição diz que $[t]_{\approx} \in R^{M/_{\approx}}$ se e somente se $M \vDash R(t)$. Se, para algum outro termo t' com $t \approx t'$, $M \nvDash R(t)$, então a definição exigiria $[t']_{\approx} \notin R^{M/_{\approx}}$. Se $t \approx t'$, então $[t]_{\approx} = [t']_{\approx}$, mas não podemos ter ao mesmo tempo $[t]_{\approx} \in R^{M/_{\approx}}$ e $[t]_{\approx} \notin R^{M/_{\approx}}$. Entretanto, Proposição 12.14 garante que isso não pode acontecer.

Proposição 12.17. *$M/_{\approx}$ está bem definida, isto é, se t_1, …, t_n, t'_1, …, t'_n são termos fechados, e $t_i \approx t'_i$, então*

1. $[f(t_1, \ldots, t_n)]_{\approx} = [f(t'_1, \ldots, t'_n)]_{\approx}$, *isto é,*

$$f(t_1, \ldots, t_n) \approx f(t'_1, \ldots, t'_n)$$

e

2. $\boldsymbol{M} \vDash R(t_1, \ldots, t_n)$ *se e somente se* $\boldsymbol{M} \vDash R(t'_1, \ldots, t'_n)$*, isto é,*

$R(t_1, \ldots, t_n) \in \Gamma^*$ *se e somente se* $R(t'_1, \ldots, t'_n) \in \Gamma^*$.

Prova. Segue de Proposição 12.14 por indução sobre n. □

Como no caso do modelo de termos, antes de provar o lema da verdade precisamos do seguinte lema.

Lema 12.18. *Seja* $\boldsymbol{M} = \boldsymbol{M}(\Gamma^*)$*; então* $\mathrm{Val}^{M/\approx}(t) = [t]_\approx$.

Prova. A prova é similar à de Lema 12.10. □

Lema 12.19. $\boldsymbol{M}/\approx \vDash A$ *se e somente se* $A \in \Gamma^*$ *para todas as sentenças* A.

Prova. Por indução sobre A, exatamente como na prova de Lema 12.12. O único caso que precisa de atenção adicional é quando $A \equiv t = t'$.

$\boldsymbol{M}/\approx \vDash t = t'$ se e somente se $[t]_\approx = [t']_\approx$ (por definição de $\boldsymbol{M}/\approx$)
se e somente se $t \approx t'$ (por definição de $[t]_\approx$)
se e somente se $t = t' \in \Gamma^*$ (por definição de $\approx$). □

Note que, embora $\boldsymbol{M}(\Gamma^*)$ seja sempre contável e infinita, $\boldsymbol{M}/\approx$ pode ser finita, já que pode ocorrer que haja apenas finitas classes $[t]_\approx$. Isso é de se esperar, já que Γ pode conter sentenças que exigem que qualquer estrutura na qual elas sejam verdadeiras seja finita. Por exemplo, $\forall x \forall y\, x = y$ é uma sentença consistente, mas é satisfeita apenas em estruturas com um domínio que contém exatamente um elemento.

12.8 O Teorema da Completude

Vamos combinar nossos resultados: chegamos ao teorema da completude.

Teorema 12.20 (Teorema da Completude). *Seja Γ um conjunto de sentenças. Se Γ é consistente, então é satisfatível.*

Prova. Suponha que Γ seja consistente. Por Lema 12.6, existe um conjunto saturado consistente $\Gamma' \supseteq \Gamma$. Por Lema 12.8, existe um $\Gamma^* \supseteq \Gamma'$ que é consistente e completo. Como $\Gamma' \subseteq \Gamma^*$, para cada fórmula $A(x)$, Γ^* contém uma sentença da forma $\exists x\, A(x) \to A(c)$ e, assim, Γ^* é saturado. Se Γ não contém $=$, então, por Lema 12.12, $M(\Gamma^*) \vDash A$ se e somente se $A \in \Gamma^*$. Disso segue, em particular, que, para todo $A \in \Gamma$, $M(\Gamma^*) \vDash A$, de modo que Γ é satisfatível. Se Γ de fato contém $=$, então, por Lema 12.19, para todas as sentenças A, $M/_{\approx} \vDash A$ se e somente se $A \in \Gamma^*$. Em particular, $M/_{\approx} \vDash A$ para todo $A \in \Gamma$, de modo que Γ é satisfatível. □

Corolário 12.21 (Teorema da Completude, Segunda Versão). *Para todo Γ e sentenças A: se $\Gamma \vDash A$ então $\Gamma \vdash A$.*

Prova. Note que os Γ em Corolário 12.21 e Teorema 12.20 são universalmente quantificados. Para nos certificarmos de que não nos confundimos, reformulemos Teorema 12.20 usando uma variável diferente: para qualquer conjunto de sentenças Δ, se Δ é consistente, então é satisfatível. Por contraposição, se Δ não é satisfatível, então Δ é inconsistente. Usaremos isso para provar o corolário.

Suponha que $\Gamma \vDash A$. Então $\Gamma \cup \{\neg A\}$ é insatisfatível por Proposição 7.28. Tomando $\Gamma \cup \{\neg A\}$ como nosso Δ, a versão anterior de Teorema 12.20 nos dá que $\Gamma \cup \{\neg A\}$ é inconsistente. Por Proposições 10.19 e 11.19, $\Gamma \vdash A$. □

12.9 O Teorema da Compacidade

Uma consequência importante do teorema da completude é o teorema da compacidade. O teorema da compacidade afirma que, se cada subconjunto *finito* de um conjunto de sentenças é satisfatível, o conjunto inteiro é satisfatível — mesmo que o próprio conjunto seja infinito. Isso está longe de ser óbvio. Não há nada que pareça descartar, ao menos à primeira vista, a possibilidade de existirem conjuntos infinitos de sentenças que sejam contraditórios, mas em que a contradição só surge, por assim dizer, do número infinito. O teorema da compacidade diz que tal cenário pode ser descartado: não existem conjuntos infinitos insatisfatíveis de sentenças tais que cada subconjunto finito deles seja satisfatível. Como o teorema da completude, ele tem uma versão relacionada à consequência lógica: se um conjunto infinito de sentenças implica algo, então já um subconjunto finito o implica.

Definição 12.22. Um conjunto Γ de fórmulas é *finitamente satisfatível* se e somente se todo $\Gamma_0 \subseteq \Gamma$ finito é satisfatível.

Teorema 12.23 (Teorema da Compacidade). *O seguinte vale para quaisquer sentenças Γ e A:*

1. *$\Gamma \vDash A$ se e somente se existe um $\Gamma_0 \subseteq \Gamma$ finito tal que $\Gamma_0 \vDash A$.*
2. *Γ é satisfatível se e somente se é finitamente satisfatível.*

Prova. Provamos (2). Se Γ é satisfatível, então existe uma estrutura M tal que $M \vDash A$ para todo $A \in \Gamma$. É claro que essa M também satisfaz todo subconjunto finito de Γ, de modo que Γ é finitamente satisfatível.

Agora suponha que Γ seja finitamente satisfatível. Então todo subconjunto finito $\Gamma_0 \subseteq \Gamma$ é satisfatível. Pela correção (Corolários 10.31 e 11.29), todo subconjunto finito é consistente. Então o próprio Γ deve ser consistente por Proposições 10.17 e 11.17.

Pela completude (Teorema 12.20), como Γ é consistente, ele é satisfatível. □

Exemplo 12.24. Em todo modelo $\mathfrak{M}$ de uma teoria Γ, cada termo t obviamente designa um elemento de $|\mathfrak{M}|$. Podemos garantir que também seja verdade que todo elemento de $|\mathfrak{M}|$ seja designado por algum termo? Em outras palavras, existem teorias Γ cujos modelos são todos cobertos? O teorema da compacidade mostra que esse não é o caso se Γ tem modelos infinitos. Eis como ver isso: Seja $\mathfrak{M}$ um modelo infinito de Γ, e seja c um símbolo de constante que não está na linguagem de Γ. Seja Δ o conjunto de todas as sentenças $c \neq t$ para t um termo na linguagem $\mathscr{L}$ de Γ, isto é,

$$\Delta = \{c \neq t : t \in \mathrm{Trm}(\mathscr{L})\}.$$

Um subconjunto finito de $\Gamma \cup \Delta$ pode ser escrito como $\Gamma' \cup \Delta'$, com $\Gamma' \subseteq \Gamma$ e $\Delta' \subseteq \Delta$. Como Δ' é finito, ele pode conter apenas finitos termos. Seja $a \in |\mathfrak{M}|$ um elemento de $|\mathfrak{M}|$ que não é designado por nenhum deles, e seja $\mathfrak{M}'$ a estrutura que é exatamente como $\mathfrak{M}$, mas com $c^{\mathfrak{M}'} = a$ também. Como $a \neq \mathrm{Val}^{\mathfrak{M}}(t)$ para todo t que ocorre em Δ', $\mathfrak{M}' \vDash \Delta'$. Como $\mathfrak{M} \vDash \Gamma$, $\Gamma' \subseteq \Gamma$, e c não ocorre em Γ, também $\mathfrak{M}' \vDash \Gamma'$. Juntando, $\mathfrak{M}' \vDash \Gamma' \cup \Delta'$ para todo subconjunto finito $\Gamma' \cup \Delta'$ de $\Gamma \cup \Delta$. Assim, todo subconjunto finito de $\Gamma \cup \Delta$ é satisfatível. Por compacidade, o próprio $\Gamma \cup \Delta$ é satisfatível. Logo, existem modelos $\mathfrak{M} \vDash \Gamma \cup \Delta$. Todo tal $\mathfrak{M}$ é um modelo de Γ, mas não é coberto, já que $\mathrm{Val}^{\mathfrak{M}}(c) \neq \mathrm{Val}^{\mathfrak{M}}(t)$ para todos os termos t de $\mathscr{L}$.

Exemplo 12.25. Considere uma linguagem $\mathscr{L}$ contendo o predicado $<$, símbolos de constante 0, 1, e símbolos de função $+$, $\times$, e $-$. Seja Γ o conjunto de todas as sentenças nesta linguagem verdadeiras na estrutura $\mathfrak{Q}$ com domínio $\mathbb{Q}$ e as interpretações óbvias. Γ é o conjunto de todas as sentenças de $\mathscr{L}$ verdadeiras a respeito dos números racionais. É claro que, em $\mathbb{Q}$ (e mesmo em $\mathbb{R}$), não há números r que sejam maiores que 0 mas menores que $1/k$ para todo $k \in \mathbb{Z}^+$. Tal número, se existisse, seria um

infinitesimal: não nulo, mas infinitamente pequeno. O teorema da compacidade pode ser usado para mostrar que há modelos de Γ nos quais infinitesimais existem. Não temos um símbolo de função para a divisão em nossa linguagem (a divisão por zero é indefinida, e símbolos de função têm de ser interpretadas por funções totais). Contudo, ainda podemos expressar que $r < 1/k$, já que isso ocorre se e somente se $r \cdot k < 1$. Agora seja c um símbolo de constante nova e seja Δ

$$\{0 < c\} \cup \{c \times \overline{k} < 1 : k \in \mathbb{Z}^+\}$$

(onde $\overline{k} = (1 + (1 + \cdots + (1 + 1)\ldots))$ com k 1's). Para qualquer subconjunto finito Δ_0 de Δ existe um K tal que todas as sentenças $c \times \overline{k} < 1$ em Δ_0 têm $k < K$. Se expandirmos $\mathbf{Q}$ para $\mathbf{Q}'$ com $c^{\mathbf{Q}'} = 1/K$ temos que $\mathbf{Q}' \vDash \Gamma_0 \cup \Delta_0$ para qualquer $\Gamma_0 \subseteq \Gamma$ finito, e assim $\Gamma \cup \Delta$ é finitamente satisfatível (Exercício: prove isso em detalhe). Por compacidade, $\Gamma \cup \Delta$ é satisfatível. Qualquer modelo $\mathbf{S}$ de $\Gamma \cup \Delta$ contém um infinitesimal, a saber $c^{\mathbf{S}}$.

Exemplo 12.26. Sabemos que a lógica de primeira ordem com identidade pode expressar que o tamanho do domínio deve ter algum tamanho mínimo: A sentença $A_{\geq n}$ (que diz "existem pelo menos n objetos distintos") é verdadeira apenas em estruturas onde $|\mathbf{M}|$ tem pelo menos n objetos. Assim, se tomarmos

$$\Delta = \{A_{\geq n} : n \geq 1\}$$

então qualquer modelo de Δ deve ser infinito. Desse modo, podemos garantir que uma teoria só tenha modelos infinitos adicionando Δ a ela: os modelos de $\Gamma \cup \Delta$ são todos os modelos infinitos de Γ, e apenas eles.

Portanto, a lógica de primeira ordem pode expressar a infinitude. O teorema da compacidade mostra que ela não pode expressar a finitude, no entanto. Pois suponha que algum conjunto de sentenças Λ fosse satisfeito em todas as estruturas finitas, e apenas nelas. Então $\Delta \cup \Lambda$ é finitamente satisfatível. Por quê? Suponha que $\Delta' \cup \Lambda' \subseteq \Delta \cup \Lambda$ seja finito, com $\Delta' \subseteq \Delta$ e $\Lambda' \subseteq \Lambda$.

Seja n o maior número tal que $A_{\geq n} \in \Delta'$. Λ, sendo satisfeito em todas as estruturas finitas, tem um modelo $\mathfrak{M}$ com um número finito de elementos, mas $\geq n$ deles. Mas então $\mathfrak{M} \models \Delta' \cup \Lambda'$. Por compacidade, $\Delta \cup \Lambda$ tem um modelo infinito, contradizendo a suposição de que Λ é satisfeito apenas em estruturas finitas.

12.10 Uma Prova Direta do Teorema da Compacidade

Podemos provar o Teorema da Compacidade diretamente, sem apelar ao Teorema da Completude, usando as mesmas ideias da prova do teorema da completude. Na prova do Teorema da Completude, começamos com um conjunto Γ consistente de sentenças, expandimos-o para um conjunto Γ^* de sentenças consistente, saturado, e completo, e então mostramos que no modelo de termos $\mathfrak{M}(\Gamma^*)$ construído a partir de Γ^*, todas as sentenças de Γ são verdadeiras, de modo que Γ é satisfatível.

Podemos usar o mesmo método para mostrar que um conjunto finitamente satisfatível de sentenças é satisfatível. Só temos que provar as versões correspondentes dos resultados que levam ao lema da verdade, substituindo "consistente" por "finitamente satisfatível".

Proposição 12.27. *Suponha que Γ é completo e finitamente satisfatível. Então:*

1. *$(A \wedge B) \in \Gamma$ se, e somente se, tanto $A \in \Gamma$ quanto $B \in \Gamma$.*
2. *$(A \vee B) \in \Gamma$ se, e somente se, $A \in \Gamma$ ou $B \in \Gamma$.*
3. *$(A \to B) \in \Gamma$ se, e somente se, $A \notin \Gamma$ ou $B \in \Gamma$.*

Lema 12.28. *Todo conjunto finitamente satisfatível Γ pode ser estendido a um conjunto saturado finitamente satisfatível Γ'.*

Proposição 12.29. *Suponha que Γ é completo, finitamente satisfatível e saturado.*

1. *$\exists x\, A(x) \in \Gamma$ se, e somente se, $A(t) \in \Gamma$ para pelo menos um termo fechado t.*
2. *$\forall x\, A(x) \in \Gamma$ se, e somente se, $A(t) \in \Gamma$ para todos os termos fechados t.*

Lema 12.30. *Todo conjunto finitamente satisfatível Γ pode ser estendido a um conjunto completo e finitamente satisfatível Γ^*.*

Teorema 12.31 (Compacidade). *Γ é satisfatível se, e somente se, é finitamente satisfatível.*

Prova. Se Γ é satisfatível, então há uma estrutura $\mathfrak{M}$ tal que $\mathfrak{M} \models A$ para todo $A \in \Gamma$. É claro que esta $\mathfrak{M}$ também satisfaz todo subconjunto finito de Γ, de modo que Γ é finitamente satisfatível.

Agora suponha que Γ é finitamente satisfatível. Por Lema 12.28, há um conjunto finitamente satisfatível e saturado $\Gamma' \supseteq \Gamma$. Por Lema 12.30, Γ' pode ser estendido a um conjunto completo e finitamente satisfatível Γ^*, e Γ^* ainda é saturado. Construa o modelo de termos $\mathfrak{M}(\Gamma^*)$ como em Definição 12.9. Note que Proposição 12.11 não dependeu do fato de que Γ^* é consistente (ou completo ou saturado, aliás), mas apenas do fato de que $\mathfrak{M}(\Gamma^*)$ é coberta. A prova do Lema da Verdade (Lema 12.12) funciona se substituirmos referências a Proposição 12.2 e Proposição 12.7 por referências a Proposição 12.27 e Proposição 12.29□

12.11 O Teorema de Löwenheim–Skolem

O Teorema de Löwenheim–Skolem diz que se uma teoria tem um modelo infinito, então ela também tem um modelo que é no máximo infinito contável. Uma consequência imediata desse

fato é que a lógica de primeira ordem não pode expressar que o tamanho de uma estrutura é incontável: qualquer sentença ou conjunto de sentenças satisfeito em todas as estruturas incontáveis também é satisfeito em alguma estrutura contável.

Teorema 12.32. *Se Γ é consistente, então ela tem um modelo contável, isto é, é satisfatível em uma estrutura cujo domínio é finito ou infinito contável.*

Prova. Se Γ é consistente, a estrutura $\boldsymbol{M}$ fornecida pela prova do teorema da completude tem um domínio $|\boldsymbol{M}|$ que não é maior do que o conjunto dos termos da linguagem $\mathscr{L}$. Assim $\boldsymbol{M}$ é no máximo infinito contável. □

Teorema 12.33. *Se Γ é um conjunto consistente de sentenças na linguagem da lógica de primeira ordem sem identidade, então ela tem um modelo infinito contável, isto é, é satisfatível em uma estrutura cujo domínio é infinito e contável.*

Prova. Se Γ é consistente e não contém sentenças em que a identidade aparece, então a estrutura $\boldsymbol{M}$ fornecida pela prova do teorema da completude tem um domínio $|\boldsymbol{M}|$ idêntico ao conjunto dos termos da linguagem $\mathscr{L}'$. Assim $\boldsymbol{M}$ é infinito contável, pois $\mathrm{Trm}(\mathscr{L}')$ o é. □

Exemplo 12.34 (Paradoxo de Skolem). A teoria dos conjuntos de Zermelo–Fraenkel **ZFC** é um arcabouço muito poderoso no qual praticamente todas as afirmações matemáticas podem ser expressas, incluindo fatos sobre os tamanhos de conjuntos. Assim, por exemplo, **ZFC** pode provar que o conjunto $\mathbb{R}$ dos números reais é incontável, pode provar o Teorema de Cantor de que o conjunto potência de qualquer conjunto é maior do que o próprio conjunto, etc. Se **ZFC** é consistente, seus modelos são todos infinitos e, além disso, todos contêm elementos sobre os quais a teoria diz que são incontáveis, como o elemento que torna verdadeiro o teorema de **ZFC** de que o conjunto potência dos números naturais existe. Pelo Teorema de Löwenheim–Skolem, **ZFC**

também tem modelos contáveis—modelos que contêm conjuntos "incontáveis", mas que eles mesmos são contáveis.

Resumo

O **teorema da completude** é a recíproca do **teorema da correção**. Em uma forma, ele afirma que se $\Gamma \models A$, então $\Gamma \vdash A$; em outra, que se Γ é consistente, então é satisfatível. Provamos a segunda forma (e derivamos a primeira a partir da segunda). A prova é trabalhosa e requer vários passos. Começamos com um conjunto consistente Γ. Primeiro, acrescentamos infinitos novos símbolos de constante c_i, bem como fórmulas da forma $\exists x\, A(x) \rightarrow A(c)$, em que cada fórmula $A(x)$ com uma variável livre na linguagem expandida é emparelhada com uma das novas constantes. Isso resulta em um conjunto consistente **saturado** de sentenças contendo Γ. Ele ainda é consistente. Agora tomamos esse conjunto e o estendemos a um **conjunto consistente completo**. Um conjunto consistente completo tem a agradável propriedade de que, para qualquer sentença A, ou A ou $\neg A$ está no conjunto (mas nunca ambos). Como partimos de um conjunto saturado, temos agora um conjunto de sentenças Γ^* saturado, completo e consistente que inclui Γ. A partir desse conjunto, é agora possível definir uma estrutura $\mathfrak{M}$ tal que $\mathfrak{M}(\Gamma^*) \models A$ se e somente se $A \in \Gamma^*$. Em particular, $\mathfrak{M}(\Gamma^*) \models \Gamma$, isto é, Γ é satisfatível. Se $=$ está presente, a construção é ligeiramente mais complexa.

Dois corolários importantes seguem do teorema da completude. O **teorema da compacidade** afirma que $\Gamma \models A$ se e somente se $\Gamma_0 \models A$ para algum $\Gamma_0 \subseteq \Gamma$ finito. Uma formulação equivalente é que Γ é satisfatível se e somente se todo $\Gamma_0 \subseteq \Gamma$ finito é satisfatível. O teorema da compacidade é útil para provar a existência de estruturas com certas propriedades. Por exemplo, podemos usá-lo para mostrar que há modelos infinitos para toda teoria que tem modelos finitos arbitrariamente grandes. Isso significa, em particular, que a finitude não pode ser expressa na

lógica de primeira ordem. O segundo corolário, o **Teorema de Löwenheim-Skolem**, afirma que todo Γ satisfatível tem um modelo contável. Ele, por sua vez, mostra que a incontabilidade não pode ser expressa na lógica de primeira ordem.

Problemas

Problema 12.1. Complete a prova de Proposição 12.2.

Problema 12.2. Complete a prova de Proposição 12.11.

Problema 12.3. Complete a prova de Lema 12.12.

Problema 12.4. Complete a prova de Proposição 12.14.

Problema 12.5. Complete a prova de Lema 12.18.

Problema 12.6. Use Corolário 12.21 para provar Teorema 12.20, mostrando assim que as duas formulações do teorema da completude são equivalentes.

Problema 12.7. Para que um sistema de derivação seja completo, suas regras devem ser fortes o suficiente para provar que todo conjunto insatisfatível é inconsistente. Quais das regras de derivação foram necessárias para provar a completude? Alguma dessas regras não é usada em nenhum lugar da prova? Para responder a essas perguntas, faça uma lista ou diagrama que mostre quais das regras de derivação foram usadas em quais resultados que levam à prova de Teorema 12.20. Certifique-se de anotar quaisquer usos tácitos de regras nessas provas.

Problema 12.8. Prove (1) de Teorema 12.23.

Problema 12.9. No modelo padrão da aritmética $\mathbf{N}$, não há elemento $k \in |\mathbf{N}|$ que satisfaça toda fórmula $\overline{n} < x$ (onde $\overline{n}$ é $0'^{\cdots'}$ com n $'$'s). Use o teorema da compacidade para mostrar que o

conjunto de sentenças na linguagem da aritmética que são verdadeiras no modelo padrão da aritmética $\mathbf{N}$ também são verdadeiras em uma estrutura $\mathbf{N}'$ que contém um elemento que *de fato* satisfaz toda fórmula $\overline{n} < x$.

Problema 12.10. Prove Proposição 12.27. Evite o uso de $\vdash$.

Problema 12.11. Prove Lema 12.28. (Dica: o passo crucial é mostrar que se Γ_n é finitamente satisfatível, então $\Gamma_n \cup \{D_n\}$ também o é, sem qualquer apelo a derivações ou consistência.)

Problema 12.12. Prove Proposição 12.29.

Problema 12.13. Prove Lema 12.30. (Dica: o passo crucial é mostrar que se Γ_n é finitamente satisfatível, então ou $\Gamma_n \cup \{A_n\}$ ou $\Gamma_n \cup \{\neg A_n\}$ é finitamente satisfatível.)

Problema 12.14. Escreva a prova completa do Lema da Verdade (Lema 12.12) na versão necessária para a prova de Teorema 12.31.

CAPÍTULO 13

Além da lógica de primeira ordem

13.1 Visão Geral

A lógica de primeira ordem não é o único sistema de lógica de interesse: há muitas extensões e variações da lógica de primeira ordem. Uma lógica normalmente consiste na especificação formal de uma linguagem, geralmente, mas nem sempre, em um sistema dedutivo e, geralmente, mas nem sempre, em uma semântica pretendida. Mas o uso técnico do termo levanta uma pergunta óbvia: o que as lógicas que não são lógicas de primeira ordem têm a ver com a palavra "lógica", usada no sentido intuitivo ou filosófico? Todos os sistemas descritos abaixo são projetados para modelar o raciocínio de uma forma ou de outra; podemos dizer o que os torna lógicos?

Não há respostas fáceis. A palavra "lógica" é usada de diferentes maneiras e em diferentes contextos, e a noção, como a de "verdade", tem sido analisada a partir de numerosos posicionamentos filosóficos. Por exemplo, pode-se considerar que

o objetivo do raciocínio lógico é a determinação de quais afirmações são necessariamente verdadeiras, verdadeiras a priori, verdadeiras independentemente da interpretação dos termos não lógicos, verdadeiras em virtude de sua forma, ou verdadeiras por convenção linguística; e cada uma dessas concepções requer bastante esclarecimento. Mesmo que se restrinja a atenção ao tipo de lógica usada na matemática, há pouco acordo quanto ao seu alcance. Por exemplo, no *Principia Mathematica*, Russell e Whitehead tentaram desenvolver a matemática com base na lógica, na tradição *logicista* iniciada por Frege. Seu sistema de lógica era uma forma de lógica de tipo superior semelhante à descrita abaixo. No final, foram forçados a introduzir axiomas que, pela maioria dos padrões, não parecem puramente lógicos (notavelmente, o axioma do infinito e o axioma da redutibilidade), mas pode-se, ainda assim, sustentar que algumas formas de raciocínio de ordem superior devem ser aceitas como lógicas. Em contraste, Quine, cuja ontologia não admite "proposições" como objetos legítimos de discurso, argumenta que a lógica de segunda ordem e a de ordem superior são realmente manifestações da teoria dos conjuntos em pele de cordeiro; em outras palavras, sistemas que envolvem quantificação sobre predicados não são puramente lógicos.

Por enquanto, é melhor deixar tais questões filosóficas para um dia chuvoso e simplesmente pensar nos sistemas abaixo como idealizações formais de vários tipos de raciocínio, lógico ou não.

13.2 Lógica de Múltiplas Ordenações

Na lógica de primeira ordem, variáveis e quantificadores variam sobre um único domínio. Mas é frequentemente útil ter vários domínios (disjuntos): por exemplo, pode-se querer ter um domínio de números, um domínio de objetos geométricos, um domínio de funções de números para números, um domínio de grupos abelianos, e assim por diante.

A lógica de múltiplas ordenações fornece esse tipo de arcabouço. Começa-se com uma lista de "ordenações"—a "ordenação" de um objeto indica o "domínio" que ele deveria habitar. Tem-se então variáveis e quantificadores para cada ordenação e (geralmente) uma identidade para cada ordenação. Funções e relações também são "tipadas" pelas ordenações dos objetos que podem receber como argumentos. Caso contrário, mantêm-se as regras usuais da lógica de primeira ordem, com versões das regras de quantificadores repetidas para cada ordenação.

Por exemplo, para estudar relações internacionais, poderíamos escolher uma linguagem com dois tipos de objetos, cidadãos franceses e cidadãos alemães. Poderíamos ter uma relação unária, "bebe vinho", para objetos da primeira ordenação; outra relação unária, "come wurst", para objetos da segunda ordenação; e uma relação binária, "forma um casal multinacional", que recebe dois argumentos, onde o primeiro argumento é da primeira ordenação e o segundo argumento é da segunda ordenação. Se usarmos variáveis a, b, c para variar sobre cidadãos franceses e x, y, z para variar sobre cidadãos alemães, então

$$\forall a \, \forall x [(CasadoCom(a,x) \rightarrow (BebeVinho(a) \vee \neg ComeWurst(x))]]$$

afirma que, se algum francês for casado com um alemão, ou o francês bebe vinho ou o alemão não come wurst.

A lógica de múltiplas ordenações pode ser embutida na lógica de primeira ordem de maneira natural, reunindo todos os objetos dos domínios de múltiplas ordenações em um único domínio de primeira ordem, usando predicados unários para acompanhar as ordenações e relativizando quantificadores. Por exemplo, a linguagem de primeira ordem correspondente ao exemplo acima teria predicados unários "$Alemao$" e "$Frances$", além das outras relações descritas, com os requisitos de ordenação eliminados. Um quantificador ordenado $\forall x \, A$, onde x é uma variável da ordenação alemã, traduz-se para

$$\forall x \, (Alemao(x) \rightarrow A).$$

Precisamos adicionar axiomas que garantam que as ordenações sejam separadas—por exemplo, $\forall x \neg(Alemao(x) \wedge Frances(x))$—, bem como axiomas que garantam que "bebe vinho" só vale de objetos que satisfazem o predicado $Frances(x)$, etc. Com essas convenções e axiomas, não é difícil mostrar que sentenças de múltiplas ordenações traduzem-se em sentenças de primeira ordem, e derivações de múltiplas ordenações traduzem-se em derivações de primeira ordem. Além disso, estruturas de múltiplas ordenações "traduzem-se" em estruturas de primeira ordem correspondentes e vice-versa, de modo que também temos um teorema de completude para a lógica de múltiplas ordenações.

13.3 Lógica de Segunda Ordem

A linguagem da lógica de segunda ordem permite quantificar não apenas sobre um domínio de indivíduos, mas também sobre relações nesse domínio. Dada uma linguagem de primeira ordem $\mathscr{L}$, para cada k adiciona-se variáveis R que variam sobre relações k-árias, e permite-se quantificação sobre essas variáveis. Se R é uma variável para uma relação k-ária, e $t_1, \ldots, t_k$ são termos ordinários (de primeira ordem), $R(t_1, \ldots, t_k)$ é uma fórmula atômica. Caso contrário, o conjunto de fórmulas é definido exatamente como no caso da lógica de primeira ordem, com cláusulas adicionais para quantificação de segunda ordem. Note que temos apenas a identidade para termos de primeira ordem: se R e S são variáveis de relação de mesma aridade k, podemos definir $R = S$ como uma abreviação de

$$\forall x_1 \ldots \forall x_k \, (R(x_1, \ldots, x_k) \leftrightarrow S(x_1, \ldots, x_k)).$$

As regras para a lógica de segunda ordem simplesmente estendem as regras de quantificadores às novas variáveis de segunda ordem. Aqui, porém, é preciso ter um pouco de cuidado para explicar como essas variáveis interagem com os predicados de $\mathscr{L}$, e com fórmulas de $\mathscr{L}$ de modo mais geral. No mínimo,

variáveis de relação contam como termos, de modo que temos inferências da forma

$$A(R) \vdash \exists R\, A(R)$$

Mas se $\mathscr{L}$ é a linguagem da aritmética com um símbolo de relação constante $<$, também esperaríamos que a seguinte inferência fosse válida:

$$x < y \vdash \exists R\, R(x,y)$$

ou, para uma dada fórmula A,

$$A(x_1,\ldots,x_k) \vdash \exists R\, R(x_1,\ldots,x_k)$$

Mais geralmente, podemos querer permitir inferências da forma

$$A[\lambda \vec{x}.\, B(\vec{x})/R] \vdash \exists R\, A$$

onde $A[\lambda \vec{x}.\, B(\vec{x})/R]$ denota o resultado de substituir cada fórmula atômica da forma $Rt_1,\ldots,t_k$ em A por $B(t_1,\ldots,t_k)$. Esta última regra é equivalente a ter um *esquema de compreensão*, isto é, um axioma da forma

$$\exists R\, \forall x_1,\ldots,x_k\, (A(x_1,\ldots,x_k) \leftrightarrow R(x_1,\ldots,x_k)),$$

um para cada fórmula A na linguagem de segunda ordem, em que R não é uma variável livre. (Exercício: mostre que se R for permitida em A, esse esquema é inconsistente!)

Quando lógicos se referem aos "axiomas da lógica de segunda ordem", geralmente querem dizer a extensão mínima da lógica de primeira ordem pelas regras de quantificadores de segunda ordem juntamente com o esquema de compreensão. Mas é frequentemente interessante estudar subsistemas mais fracos desses axiomas e regras. Por exemplo, note que em toda a sua generalidade o esquema de axiomas de compreensão é *impredicativo*: ele permite afirmar a existência de uma relação $R(x_1,\ldots,x_k)$ que é "definida" por uma fórmula com quantificadores de segunda ordem; e esses quantificadores variam sobre o conjunto de todas

essas relações—um conjunto que inclui R em si! Por volta da virada do século XX, uma reação comum ao paradoxo de Russell foi atribuir a culpa a tais definições e evitá-las ao desenvolver os fundamentos da matemática. Se se proíbe o uso de quantificadores de segunda ordem na fórmula A, tem-se uma forma *predicativa* de compreensão, que é um pouco mais fraca.

Do ponto de vista semântico, pode-se pensar em uma estrutura de segunda ordem como consistindo em uma estrutura de primeira ordem para a linguagem, juntamente com um conjunto de relações no domínio sobre o qual variem os quantificadores de segunda ordem (mais precisamente, para cada k há um conjunto de relações de aridade k). É claro que, se a compreensão estiver incluída no sistema de derivação, então temos o requisito adicional de que há relações suficientes na "parte de segunda ordem" para satisfazer os axiomas de compreensão—caso contrário, o sistema de derivação não é correto! Uma maneira fácil de garantir que há relações suficientes é tomar a parte de segunda ordem como consistindo de *todas* as relações sobre a parte de primeira ordem. Uma tal estrutura é chamada *plena*, e, de certo modo, é realmente a "estrutura pretendida" para a linguagem. Se restringirmos a atenção a estruturas plenas, temos o que se conhece como a semântica de segunda ordem *plena*. Nesse caso, especificar uma estrutura reduz-se a especificar a parte de primeira ordem, pois o conteúdo da parte de segunda ordem segue disso implicitamente.

Para resumir, há certa ambiguidade ao falar de lógica de segunda ordem. Em termos do sistema de derivação, pode-se ter em mente

1. Um sistema de derivação de segunda ordem "mínimo", juntamente com alguns axiomas de compreensão.

2. O sistema de derivação de segunda ordem "padrão", com compreensão plena.

Em termos da semântica, pode interessar

1. A semântica "fraca", em que uma estrutura consiste em uma parte de primeira ordem, juntamente com uma parte de segunda ordem grande o suficiente para satisfazer os axiomas de compreensão.

2. A semântica de segunda ordem "padrão", em que se consideram apenas estruturas plenas.

Quando lógicos não especificam o sistema de derivação ou a semântica que têm em mente, geralmente se referem ao segundo item de cada lista. A vantagem de usar essa semântica é que, como veremos, ela nos dá descrições categóricas de muitas estruturas matemáticas naturais; ao mesmo tempo, o sistema de derivação é bastante forte, e correto para essa semântica. A desvantagem é que o sistema de derivação *não* é completo para a semântica; de fato, *nenhum* sistema de derivação efetivamente dado é completo para a semântica plena de segunda ordem. Por outro lado, veremos que o sistema de derivação *é* completo para a semântica enfraquecida; isso implica que, se uma sentença não for demonstrável, então há *alguma* estrutura, não necessariamente a plena, na qual ela é falsa.

A linguagem da lógica de segunda ordem é bastante rica. Pode-se identificar relações unárias com subconjuntos do domínio, e assim, em particular, pode-se quantificar sobre esses conjuntos; por exemplo, pode-se expressar indução para os números naturais com um único axioma

$$\forall R\,((R(0) \wedge \forall x\,(R(x) \to R(x'))) \to \forall x\,R(x)).$$

Se se toma a linguagem da aritmética com os símbolos $0, ', +, \times$ e $<$, pode-se adicionar os seguintes axiomas para descrever seu comportamento:

1. $\forall x\,\neg x' = 0$

2. $\forall x\,\forall y\,(s(x) = s(y) \to x = y)$

3. $\forall x\,(x + 0) = x$

4. $\forall x \forall y\, (x + y') = (x + y)'$
5. $\forall x\, (x \times 0) = 0$
6. $\forall x \forall y\, (x \times y') = ((x \times y) + x)$
7. $\forall x \forall y\, (x < y \leftrightarrow \exists z\, y = (x + z'))$

Não é difícil mostrar que esses axiomas, juntamente com o axioma de indução acima, fornecem uma descrição categórica da estrutura $\boldsymbol{N}$, o modelo padrão da aritmética, desde que estejamos usando a semântica plena de segunda ordem. Dada qualquer estrutura $\boldsymbol{M}$ na qual esses axiomas são verdadeiros, defina uma função f de $\mathbb{N}$ para o domínio de $\boldsymbol{M}$ usando recursão ordinária em $\mathbb{N}$, de modo que $f(0) = 0^{\boldsymbol{M}}$ e $f(x + 1) = '^{\boldsymbol{M}}(f(x))$. Usando indução ordinária em $\mathbb{N}$ e o fato de que os axiomas (1) e (2) valem em $\boldsymbol{M}$, vemos que f é injetora. Para ver que f é sobrejetora, seja P o conjunto de elementos de $|\boldsymbol{M}|$ que estão no alcance de f. Como $\boldsymbol{M}$ é plena, P está no domínio de segunda ordem. Pela construção de f, sabemos que $0^{\boldsymbol{M}}$ está em P, e que P é fechado sob $'^{\boldsymbol{M}}$. O fato de que o axioma de indução vale em $\boldsymbol{M}$ (em particular, para P) garante que P é igual a todo o domínio de primeira ordem de $\boldsymbol{M}$. Isso mostra que f é uma bijeção. Mostrar que f é um homomorfismo não é mais difícil, usando indução ordinária em $\mathbb{N}$ repetidamente.

Em termos teórico-conjuntistas, uma função é apenas um tipo especial de relação; por exemplo, uma função unária f pode ser identificada com uma relação binária R que satisfaz $\forall x \exists! y\, R(x, y)$. Como resultado, pode-se também quantificar sobre funções. Usando a semântica plena, pode-se então definir a classe de estruturas infinitas como a classe de estruturas $\boldsymbol{M}$ para as quais há uma função injetiva do domínio de $\boldsymbol{M}$ para um subconjunto próprio de si mesma:

$$\exists f\, (\forall x \forall y\, (f(x) = f(y) \rightarrow x = y) \wedge \exists y \forall x\, f(x) \neq y).$$

A negação dessa sentença define então a classe de estruturas finitas.

Além disso, pode-se definir a classe de ordens bem-fundadas, adicionando o seguinte à definição de uma ordenação linear:

$$\forall P\,(\exists x\,P(x) \rightarrow \exists x\,(P(x) \wedge \forall y\,(y < x \rightarrow \neg P(y)))).$$

Isso afirma que todo conjunto não vazio tem um elemento mínimo, módulo a identificação de "conjunto" com "relação unária". Como outro exemplo, pode-se expressar a noção de conexidade para grafos, dizendo que não há separação não trivial dos vértices em partes desconectadas:

$$\neg\exists A\,(\exists x\,A(x) \wedge \exists y\,\neg A(y) \wedge \forall w\,\forall z\,((A(w) \wedge \neg A(z)) \rightarrow \neg R(w,z))).$$

Como mais um exemplo, pode-se tentar, como exercício, definir a classe de estruturas finitas cujo domínio tem tamanho par. De modo ainda mais impressionante, pode-se fornecer uma descrição categórica dos números reais como um corpo ordenado completo que contém os racionais.

Em suma, a lógica de segunda ordem é muito mais expressiva do que a lógica de primeira ordem. Essa é a boa notícia; agora a má. Já mencionamos que não há sistema de derivação efetivo que seja completo para a semântica plena de segunda ordem. Para o bem ou para o mal, muitas das propriedades da lógica de primeira ordem estão ausentes, incluindo compacidade e os teoremas de Löwenheim–Skolem.

Por outro lado, se se está disposto a abrir mão da semântica plena de segunda ordem em favor da mais fraca, então o sistema de derivação mínimo de segunda ordem é completo para essa semântica. Em outras palavras, se lemos $\vdash$ como "demonstra no sistema mínimo" e $\models$ como "implica logicamente na semântica fraca", podemos mostrar que sempre que $\Gamma \models A$, então $\Gamma \vdash A$. Se se quer incluir axiomas específicos de compreensão no sistema de derivação, é preciso restringir a semântica a estruturas de segunda ordem que satisfaçam esses axiomas: por exemplo, se Δ consiste em um conjunto de axiomas de compreensão (possivelmente todos eles), temos que, se $\Gamma \cup \Delta \models A$, então $\Gamma \cup \Delta \vdash A$. Em particular, se A não for demonstrável usando os axiomas de

compreensão que estamos considerando, então há um modelo de $\neg A$ no qual esses axiomas de compreensão ainda assim valem.

A maneira mais fácil de ver que o teorema de completude vale para a semântica fraca é pensar na lógica de segunda ordem como uma lógica de múltiplas ordenações, da seguinte forma. Uma ordenação é interpretada como o domínio ordinário de "primeira ordem", e então, para cada k, temos um domínio de "relações de aridade k." Tomamos a linguagem com símbolos de relação incorporados "$verd_k(R, x_1, \ldots, x_k)$", destinados a afirmar que R vale de x_1, ..., x_k, onde R é uma variável da ordenação "relação k-ária" e x_1, ..., x_k são objetos da ordenação de primeira ordem.

Com essa identificação, a semântica fraca de segunda ordem é essencialmente a semântica usual da lógica de múltiplas ordenações; e já observamos que a lógica de múltiplas ordenações pode ser embutida na lógica de primeira ordem. Módulo as traduções de ida e volta, então, a concepção mais fraca de lógica de segunda ordem é realmente uma forma disfarçada de lógica de primeira ordem, em que o domínio contém tanto "objetos" quanto "relações" governados pelos axiomas apropriados.

13.4 Lógica de Ordem Superior

Passar da lógica de primeira ordem para a de segunda ordem nos permitiu falar sobre conjuntos de objetos no domínio de primeira ordem, dentro da linguagem formal. Por que parar aí? Por exemplo, a lógica de terceira ordem deveria nos permitir lidar com conjuntos de conjuntos de objetos, ou talvez até conjuntos que contenham tanto objetos quanto conjuntos de objetos. E a lógica de quarta ordem nos permitirá falar sobre conjuntos de objetos desse tipo. Como se pode imaginar, pode-se iterar essa ideia arbitrariamente.

Na prática, a lógica de ordem superior é frequentemente formulada em termos de funções em vez de relações. (Módulo as identificações naturais, essa diferença é inessencial.) Dadas al-

gumas "ordenações" básicas A, B, C, ... (que agora chamaremos de "tipos"), podemos criar novos estipulando

Se σ e τ são tipos finitos, então $\sigma \to \tau$ também o é.

Pense nos tipos como "rótulos" sintáticos que classificam os objetos que queremos em nosso domínio; $\sigma \to \tau$ descreve aqueles objetos que são funções que levam objetos de tipo σ a objetos de tipo τ. Por exemplo, podemos querer ter um tipo Ω de valores de verdade, "verdadeiro" e "falso", e um tipo $\mathbb{N}$ de números naturais. Nesse caso, pode-se pensar em objetos de tipo $\mathbb{N} \to \Omega$ como relações unárias, ou subconjuntos de $\mathbb{N}$; objetos de tipo $\mathbb{N} \to \mathbb{N}$ são funções de números naturais para números naturais; e objetos de tipo $(\mathbb{N} \to \mathbb{N}) \to \mathbb{N}$ são "funcionais", isto é, funções de tipo superior que levam funções a números.

Como no caso da lógica de segunda ordem, pode-se pensar na lógica de ordem superior como uma espécie de lógica de múltiplas ordenações, em que há uma ordenação para cada tipo de objeto que queremos considerar. Mas é usualmente mais claro simplesmente definir a sintaxe da lógica de tipos superiores a partir dos fundamentos. Por exemplo, podemos definir um conjunto de tipos finitos indutivamente, da seguinte forma:

1. $\mathbb{N}$ é um tipo finito.
2. Se σ e τ são tipos finitos, então $\sigma \to \tau$ também o é.
3. Se σ e τ são tipos finitos, então $\sigma \times \tau$ também o é.

Intuitivamente, $\mathbb{N}$ denota o tipo dos números naturais, $\sigma \to \tau$ denota o tipo de funções de σ para τ, e $\sigma \times \tau$ denota o tipo de pares de objetos, um de σ e outro de τ. Podemos então definir um conjunto de termos indutivamente, da seguinte forma:

1. Para cada tipo σ, há um estoque de variáveis x, y, z, ... de tipo σ
2. 0 é um termo de tipo $\mathbb{N}$

3. S (sucessor) é um termo de tipo $\mathbb{N} \to \mathbb{N}$

4. Se s é um termo de tipo σ, e t é um termo de tipo $\mathbb{N} \to (\sigma \to \sigma)$, então R_{st} é um termo de tipo $\mathbb{N} \to \sigma$

5. Se s é um termo de tipo $\tau \to \sigma$ e t é um termo de tipo τ, então $s(t)$ é um termo de tipo σ

6. Se s é um termo de tipo σ e x é uma variável de tipo τ, então $\lambda x.\, s$ é um termo de tipo $\tau \to \sigma$.

7. Se s é um termo de tipo σ e t é um termo de tipo τ, então $\langle s, t \rangle$ é um termo de tipo $\sigma \times \tau$.

8. Se s é um termo de tipo $\sigma \times \tau$, então $p_1(s)$ é um termo de tipo σ e $p_2(s)$ é um termo de tipo τ.

Intuitivamente, R_{st} denota a função definida recursivamente por

$$R_{st}(0) = s$$
$$R_{st}(x+1) = t(x, R_{st}(x)),$$

$\langle s, t \rangle$ denota o par cuja primeira componente é s e cuja segunda componente é t, e $p_1(s)$ e $p_2(s)$ denotam o primeiro e o segundo elementos ("projeções") de s. Finalmente, $\lambda x.\, s$ denota a função f definida por

$$f(x) = s$$

para qualquer x de tipo σ; assim o item (6) nos dá uma forma de compreensão, permitindo-nos definir funções usando termos. As fórmulas são construídas a partir de afirmações de identidade $s = t$ entre termos do mesmo tipo, dos conectivos proposicionais usuais e da quantificação de ordem superior. Pode-se então tomar os axiomas do sistema como as equações básicas que governam os termos definidos acima, juntamente com as regras usuais da lógica com quantificadores e identidade.

Se se aumenta o sistema de tipos finitos com um tipo Ω de valores de verdade, é preciso incluir axiomas que governem seu uso também. De fato, se se for astuto, pode-se eliminar fórmulas

complexas inteiramente, substituindo-as por termos de tipo Ω! O sistema de prova pode então ser modificado de acordo. O resultado é essencialmente a *teoria simples dos tipos* apresentada por Alonzo Church na década de 1930.

Como no caso da lógica de segunda ordem, há diferentes versões de semântica de tipos superiores que se pode querer usar. Na versão completa, variáveis de tipo $\sigma \to \tau$ variam sobre o conjunto de *todas* as funções dos objetos de tipo σ para objetos de tipo τ. Como se pode imaginar, essa semântica é forte demais para admitir um sistema de derivação completo e efetivo. Mas pode-se considerar uma semântica mais fraca, em que uma estrutura consiste em conjuntos de elementos T_τ para cada tipo τ, juntamente com operações apropriadas para aplicação, projeção, etc. Se os detalhes forem desenvolvidos corretamente, pode-se obter teoremas de completude para os tipos de sistemas de derivação descritos acima.

A lógica de ordem superior é atraente porque fornece um arcabouço no qual podemos embutir boa parte da matemática de maneira natural: a partir de $\mathbb{N}$, pode-se definir números reais, funções contínuas e assim por diante. É também particularmente atraente no contexto da lógica intuicionista, pois os tipos têm interpretações "construtivas" claras. De fato, pode-se desenvolver versões construtivas da semântica de tipos superiores (baseadas na lógica intuicionista, em vez da clássica) que esclarecem bastante essas interpretações construtivas e são, em muitos aspectos, mais interessantes do que as contrapartes clássicas.

13.5 Lógica Intuicionista

Em contraste com a lógica de segunda ordem e de ordem superior, a lógica intuicionista de primeira ordem representa uma restrição da versão clássica, destinada a modelar um tipo de raciocínio mais "construtivo". Os exemplos a seguir podem ilustrar algumas das motivações subjacentes.

Suponha que alguém se aproximasse de você um dia e anunciasse que havia determinado um número natural x, com a propriedade de que se x é primo, a hipótese de Riemann é verdadeira, e se x é composto, a hipótese de Riemann é falsa. Ótimas notícias! Se a hipótese de Riemann é verdadeira ou não é uma das grandes questões em aberto da matemática, e aqui parecem ter reduzido o problema a um cálculo, isto é, à determinação de se um número específico é primo ou não.

Qual é o valor mágico de x? Eles o descrevem assim: x é o número natural que é igual a 7 se a hipótese de Riemann é verdadeira, e 9 caso contrário.

Com raiva, você exige seu dinheiro de volta. De um ponto de vista clássico, a descrição acima de fato determina um valor único de x; mas o que você realmente quer é um valor de x que seja dado *explicitamente*.

Para tomar outro exemplo, talvez menos artificial, considere a seguinte questão. Sabemos que é possível elevar um número irracional a uma potência racional e obter um resultado racional. Por exemplo, $\sqrt{2}^2 = 2$. O que é menos claro é se é possível elevar um número irracional a uma potência *irracional* e obter um resultado racional. O seguinte teorema responde afirmativamente:

Teorema 13.1. *Existem números irracionais a e b tais que a^b é racional.*

Prova. Considere $\sqrt{2}^{\sqrt{2}}$. Se isto é racional, terminamos: podemos tomar $a = b = \sqrt{2}$. Caso contrário, é irracional. Então temos

$$(\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}\cdot\sqrt{2}} = \sqrt{2}^2 = 2,$$

que é certamente racional. Assim, neste caso, tome a como $\sqrt{2}^{\sqrt{2}}$, e b como $\sqrt{2}$. □

Isso constitui uma prova válida? A maioria dos matemáticos sente que sim. Mas, novamente, há algo um pouco insatisfatório

aqui: provamos a existência de um par de números reais com uma certa propriedade, sem poder dizer *qual* par de números é. É possível provar o mesmo resultado, mas de tal forma que o par a, b *é* dado na prova: tome $a = \sqrt{3}$ e $b = \log_3 4$. Então

$$a^b = \sqrt{3}^{\log_3 4} = 3^{1/2 \cdot \log_3 4} = (3^{\log_3 4})^{1/2} = 4^{1/2} = 2,$$

pois $3^{\log_3 x} = x$.

A lógica intuicionista foi concebida para modelar um tipo de raciocínio no qual movimentos como o da primeira prova são proibidos. Provar a existência de um x que satisfaz $A(x)$ significa que se deve fornecer um x específico e uma prova de que ele satisfaz A, como na segunda prova. Provar que A ou B vale requer que se possa provar um ou outro.

Formalmente falando, a lógica intuicionista de primeira ordem é o que se obtém se se restringe um sistema de derivação para a lógica de primeira ordem de certa maneira. Similarmente, há versões intuicionistas da lógica de segunda ou de ordem superior. Do ponto de vista matemático, estes são apenas sistemas dedutivos formais, mas, como já observado, são destinados a modelar um tipo de raciocínio matemático. Pode-se tomar isso como o tipo de raciocínio justificado por uma certa visão filosófica da matemática (como o intuicionismo de Brouwer); pode-se tomá-lo como um tipo de raciocínio matemático mais "concreto" e satisfatório (na linha do construtivismo de Bishop); e pode-se discutir se a descrição formal captura ou não a motivação informal. Mas quaisquer que sejam as posições filosóficas que possamos ter, podemos estudar a lógica intuicionista como uma lógica formalmente apresentada; e por quaisquer razões, muitos lógicos matemáticos acham interessante fazê-lo.

Há uma interpretação construtiva informal dos conectivos intuicionistas, usualmente conhecida como interpretação BHK (em homenagem a Brouwer, Heyting e Kolmogorov). Ela funciona assim: uma prova de $A \wedge B$ consiste em uma prova de A emparelhada com uma prova de B; uma prova de $A \vee B$ consiste em uma prova de A, ou uma prova de B, onde temos informação explícita

sobre qual é o caso; uma prova de $A \to B$ consiste em um procedimento que transforma uma prova de A em uma prova de B; uma prova de $\forall x\, A(x)$ consiste em um procedimento que retorna uma prova de $A(x)$ para qualquer valor de x; e uma prova de $\exists x\, A(x)$ consiste em um valor de x, juntamente com uma prova de que esse valor satisfaz A. Pode-se descrever a interpretação em termos computacionais conhecidos como o "isomorfismo de Curry–Howard" ou o "paradigma fórmulas-como-tipos": pense em uma fórmula como especificando um certo tipo de dado, e provas como objetos computacionais desses tipos de dados que nos permitem ver que a fórmula correspondente é verdadeira.

A lógica intuicionista é frequentemente pensada como a lógica clássica "menos" a lei do terceiro excluído. O seguinte teorema torna isso mais preciso.

Teorema 13.2. *Intuicionisticamente, os seguintes esquemas de axiomas são equivalentes:*

1. $(\neg A \to \bot) \to A$.
2. $A \vee \neg A$
3. $\neg\neg A \to A$

Obter instâncias de um esquema a partir de qualquer um dos outros é um bom exercício em lógica intuicionista.

Os primeiros sistemas dedutivos para a lógica proposicional intuicionista, apresentados como formalizações do intuicionismo de Brouwer, são devidos, independentemente, a Kolmogorov, Glivenko e Heyting. A primeira formalização da lógica intuicionista de primeira ordem (e partes da matemática intuicionista) é devida a Heyting. Embora vários esquemas válidos clássicamente não sejam válidos intuicionisticamente, muitos o são.

A *tradução de dupla negação* descreve uma relação importante entre a lógica clássica e a intuicionista. Ela é definida indutivamente da seguinte forma (pense em A^N como a tradução "intui-

cionista" da fórmula clássica A):

$$\begin{aligned} A^N &\equiv \neg\neg A \quad \text{para fórmulas atômicas } A \\ (A \wedge B)^N &\equiv (A^N \wedge B^N) \\ (A \vee B)^N &\equiv \neg\neg(A^N \vee B^N) \\ (A \to B)^N &\equiv (A^N \to B^N) \\ (\forall x\, A)^N &\equiv \forall x\, A^N \\ (\exists x\, A)^N &\equiv \neg\neg\exists x\, A^N \end{aligned}$$

Kolmogorov e Glivenko tiveram versões dessa tradução para a lógica proposicional; para a lógica de predicados, é devida a Gödel e Gentzen, independentemente. Temos

Teorema 13.3. *1. $A \leftrightarrow A^N$ é demonstrável classicamente*

2. Se A é demonstrável classicamente, então A^N é demonstrável intuicionisticamente.

Podemos agora imaginar o seguinte diálogo. Matemático clássico: "Provei A!" Matemático intuicionista: "Sua prova não é válida. O que você realmente provou é A^N." Matemático clássico: "Tudo bem para mim!" Quanto ao matemático clássico, o intuicionista está apenas sendo minucioso, pois os dois são equivalentes. Mas o intuicionista insiste que há uma diferença.

Note que a tradução acima diz respeito apenas à lógica pura; ela não aborda a questão de quais são os axiomas *não lógicos* adequados para a matemática clássica e intuicionista, ou qual é a relação entre eles. Mas a seguinte extensão ligeira do teorema acima fornece alguma informação útil:

Teorema 13.4. *Se Γ prova A classicamente, então Γ^N prova A^N intuicionisticamente.*

Em outras palavras, se A é demonstrável a partir de algumas hipóteses classicamente, então A^N é demonstrável a partir de suas traduções de dupla negação.

Para mostrar que uma sentença ou fórmula proposicional é intuicionisticamente válida, basta fornecer uma prova. Mas como se pode mostrar que ela não é válida? Para esse propósito, precisamos de uma semântica que seja correta e, de preferência, completa. Uma semântica devida a Kripke se encaixa bem.

Podemos jogar o mesmo jogo que fizemos para a lógica clássica: definir a semântica e provar correção e completude. Vale a pena, no entanto, notar a seguinte distinção. No caso da lógica clássica, a semântica era a "óbvia", num sentido implícito no significado dos conectivos. Embora se possa fornecer alguma motivação intuitiva para a semântica de Kripke, esta não oferece o mesmo sentimento de inevitabilidade. Além disso, a noção de estrutura clássica é uma noção matemática natural, de modo que podemos ou tomar a noção de estrutura como uma ferramenta para estudar a lógica clássica de primeira ordem, ou tomar a lógica clássica de primeira ordem como uma ferramenta para estudar estruturas matemáticas. Em contraste, as estruturas de Kripke só podem ser vistas como um construto lógico; elas não parecem ter interesse matemático independente.

Uma estrutura de Kripke $\mathfrak{M} = \langle W, R, V \rangle$ para uma linguagem proposicional consiste em um conjunto W, uma ordem parcial R em W com um menor elemento, e uma atribuição "monótona" de variáveis proposicionais aos elementos de W. A intuição é que os elementos de W representam "mundos", ou "estados de conhecimento"; um elemento $v \geq u$ representa um "possível estado futuro" de u; e as variáveis proposicionais atribuídas a u são as proposições que se sabe serem verdadeiras no estado u. A relação de forçamento $\mathfrak{M}, w \Vdash A$ estende essa relação a fórmulas arbitrárias na linguagem; leia $\mathfrak{M}, w \Vdash A$ como "A é verdadeira no estado w." A relação é definida indutivamente, da seguinte forma:

1. $\mathfrak{M}, w \Vdash p_i$ se, e somente se, p_i é uma das variáveis proposicionais atribuídas a w.

2. $\mathfrak{M}, w \nVdash \bot$.

3. $\mathfrak{M}, w \Vdash (A \wedge B)$ se, e somente se, $\mathfrak{M}, w \Vdash A$ e $\mathfrak{M}, w \Vdash B$.

4. $\mathfrak{M}, w \Vdash (A \vee B)$ se, e somente se, $\mathfrak{M}, w \Vdash A$ ou $\mathfrak{M}, w \Vdash B$.

5. $\mathfrak{M}, w \Vdash (A \to B)$ se, e somente se, sempre que $w' \geq w$ e $\mathfrak{M}, w' \Vdash A$, então $\mathfrak{M}, w' \Vdash B$.

É um bom exercício tentar mostrar que $\neg(p \wedge q) \to (\neg p \vee \neg q)$ não é intuicionisticamente válida, preparando uma estrutura de Kripke que forneça um contraexemplo.

13.6 Lógicas Modais

Considere o seguinte exemplo de uma sentença condicional:

> Se Jeremy está sozinho naquela sala, então ele está bêbado, nu e dançando nas cadeiras.

Este é um exemplo de uma asserção condicional que pode ser materialmente verdadeira, mas ainda assim enganosa, pois parece sugerir que há um vínculo mais forte entre o antecedente e a conclusão do que simplesmente o fato de que ou o antecedente é falso ou a conclusão é verdadeira. Ou seja, a formulação sugere que a afirmação não é apenas verdadeira neste mundo particular (onde pode ser trivialmente verdadeira, porque Jeremy não está sozinho na sala), mas que, além disso, a conclusão *teria sido* verdadeira *se* o antecedente tivesse sido verdadeiro. Em outras palavras, pode-se tomar a asserção como significando que a afirmação é verdadeira não apenas neste mundo, mas em qualquer mundo "possível"; ou que ela é *necessariamente* verdadeira, em oposição a ser apenas verdadeira neste mundo particular.

A lógica modal foi concebida para dar sentido a esse tipo de necessidade. Obtém-se a lógica proposicional modal a partir da lógica proposicional ordinária adicionando um operador caixa; isto é, se A é uma fórmula, então $\Box A$ também o é. Intuitivamente, $\Box A$ afirma que A é *necessariamente* verdadeira, ou verdadeira em

qualquer mundo possível. $\Diamond A$ é usualmente tomada como abreviação de $\neg\Box\neg A$, e pode ser lida como afirmando que A é *possivelmente* verdadeira. É claro que a modalidade também pode ser adicionada à lógica de predicados.

Estruturas de Kripke podem ser usadas para fornecer uma semântica para a lógica modal; de fato, Kripke projetou essa semântica tendo a lógica modal em mente. Em vez de restringir a ordens parciais, de modo mais geral tem-se um conjunto de "mundos possíveis", P, e uma relação binária de "acessibilidade" $R(x,y)$ entre mundos. Intuitivamente, $R(p,q)$ afirma que o mundo q é compatível com p; isto é, se estamos "em" o mundo p, temos que considerar a possibilidade de que o mundo pudesse ter sido como q.

A lógica modal é às vezes chamada de lógica "intensional", em oposição a uma "extensional". A semântica pretendida para uma lógica extensional, como a lógica clássica, refere-se apenas a um único mundo, o "atual"; enquanto a semântica para uma lógica "intensional" depende de uma ontologia mais elaborada. Além de estruturar a necessidade, pode-se usar a modalidade para estruturar outras construções linguísticas, reinterpretando $\Box$ e $\Diamond$ de acordo com a aplicação. Por exemplo:

1. Na lógica de demonstrabilidade, $\Box A$ é lida como "A é demonstrável" e $\Diamond A$ é lida como "A é consistente".

2. Na lógica epistêmica, pode-se ler $\Box A$ como "eu sei A" ou "eu acredito em A".

3. Na lógica temporal, pode-se ler $\Box A$ como "A é sempre verdadeira" e $\Diamond A$ como "A é às vezes verdadeira".

Gostaríamos de aumentar a lógica com regras e axiomas que lidem com modalidade. Por exemplo, o sistema **S4** consiste nos axiomas e regras ordinários da lógica proposicional, juntamente

com os seguintes axiomas:

$$\Box(A \to B) \to (\Box A \to \Box B)$$
$$\Box A \to A$$
$$\Box A \to \Box\Box A$$

assim como uma regra, "de A conclua $\Box A$." **S5** adiciona o seguinte axioma:

$$\Diamond A \to \Box\Diamond A$$

Variações desses axiomas podem ser adequadas para diferentes aplicações; por exemplo, S5 é usualmente tomado para caracterizar a noção de necessidade lógica. E o aspecto agradável é que se pode usualmente encontrar uma semântica para a qual o sistema de derivação é correto e completo, restringindo a relação de acessibilidade nas estruturas de Kripke de maneiras naturais. Por exemplo, **S4** corresponde à classe de estruturas de Kripke em que a relação de acessibilidade é reflexiva e transitiva. **S5** corresponde à classe de estruturas de Kripke em que a relação de acessibilidade é *universal*, isto é, todo mundo é acessível a partir de qualquer outro; assim $\Box A$ vale se, e somente se, A vale em todo mundo.

13.7 Outras Lógicas

Como você já deve ter percebido, não é difícil projetar uma nova lógica. Você também pode criar a sua própria sintaxe, montar um sistema dedutivo e elaborar uma semântica para acompanhá-la. Talvez você tenha que ser um pouco esperto se quiser que o sistema de derivação seja completo para a semântica, e pode exigir algum esforço convencer o mundo em geral de que a sua lógica é realmente interessante. Mas, em troca, você pode desfrutar de horas de diversão sadia, explorando as propriedades matemáticas e computacionais da sua lógica.

As últimas décadas testemunharam uma verdadeira explosão de lógicas formais. A lógica difusa é projetada para modelar o raciocínio sobre propriedades vagas. A lógica probabilística é projetada para modelar o raciocínio sobre a incerteza. As lógicas *default* (de exceções) e as lógicas não monotônicas são projetadas para modelar formas revogáveis de raciocínio, isto é, inferências "razoáveis" que podem mais tarde ser derrubadas diante de novas informações. Há lógicas epistêmicas, projetadas para modelar o raciocínio sobre o conhecimento; lógicas causais, projetadas para modelar o raciocínio sobre relações causais; e até mesmo lógicas "deônticas", que são projetadas para modelar o raciocínio sobre obrigações morais e éticas. Dependendo de a principal motivação para a introdução desses sistemas ser filosófica, matemática ou computacional, você pode encontrar estudos sobre essas criaturas sob a rubrica de lógica matemática, lógica filosófica, inteligência artificial, ciência cognitiva ou outro lugar.

A lista não tem fim, e as possibilidades parecem infinitas. Talvez nunca alcancemos o sonho de Leibniz de reduzir toda a razão humana a cálculo — mas isso não pode nos impedir de tentar.

Alan Turing
1912 - 1954

PARTE III

Máquinas de Turing

CAPÍTULO 14

Computações de Máquinas de Turing

14.1 Introdução

O que significa dizer que uma função, digamos, de $\mathbb{N}$ para $\mathbb{N}$ é *computável*? Entre as primeiras respostas, e a mais conhecida, está a de que uma função é computável se puder ser computada por uma máquina de Turing. Essa noção foi apresentada por Alan Turing em 1936. As máquinas de Turing são um exemplo de *modelo de computação*—uma maneira matematicamente precisa de definir a ideia de um "procedimento computacional." O que exatamente isso significa é debatido, mas há amplo consenso de que as máquinas de Turing são uma forma de especificar procedimentos computacionais. Embora o termo "máquina de Turing" evoque a imagem de uma máquina física com partes móveis, estritamente falando uma máquina de Turing é um construto puramente matemático e, como tal, idealiza a ideia de um procedimento computacional. Por exemplo, não impomos restrição alguma aos requisitos de tempo ou memória de uma máquina

de Turing: máquinas de Turing podem computar algo mesmo que a computação exija mais espaço de armazenamento ou mais passos do que há átomos no universo.

Talvez seja melhor pensar em uma máquina de Turing como um programa para um tipo especial de mecanismo imaginário. Esse mecanismo consiste em uma *fita* e uma *cabeça de leitura-escrita*. Na nossa versão das máquinas de Turing, a fita é infinita em uma direção (à direita) e é dividida em *quadrados*, cada um dos quais pode conter um símbolo de um *alfabeto* finito. Esses alfabetos podem conter qualquer número de símbolos diferentes, mas nos contentaremos em geral com três: $\triangleright$, $\sqcup$ e I. Quando o mecanismo é iniciado, a fita está vazia (ou seja, cada quadrado contém o símbolo $\sqcup$), exceto o quadrado mais à esquerda, que contém $\triangleright$, e um número finito de quadrados que contêm a *entrada*. A qualquer momento, o mecanismo está em um de um número finito de *estados*. No início, a cabeça examina o quadrado mais à esquerda e em um *estado inicial* especificado. A cada passo da execução do mecanismo, o conteúdo do quadrado atualmente examinado, junto com o estado em que o mecanismo se encontra e o programa da máquina de Turing, determinam o que acontece em seguida. O programa da máquina de Turing é dado por uma função parcial que recebe como entrada um estado q e um símbolo σ e produz uma tripla $\langle q', \sigma', D \rangle$. Sempre que o mecanismo está no estado q e lê o símbolo σ, substitui o símbolo no quadrado atual por σ', a cabeça move-se à esquerda, à direita ou permanece no lugar conforme D seja L, R ou N, e o mecanismo passa ao estado q'.

Por exemplo, considere a situação em Figura 14.1. A parte visível da fita da máquina de Turing contém o símbolo de fim de fita $\triangleright$ no quadrado mais à esquerda, seguido de três 1's, um 0 e mais quatro 1's. A cabeça está lendo o terceiro quadrado da esquerda, que contém um 1, e está no estado q_1—dizemos que "a máquina está lendo um 1 no estado q_1." Se o programa da máquina de Turing retornar, para a entrada $\langle q_1, 1 \rangle$, a tripla $\langle q_2, 0, N \rangle$, a máquina substituiria o 1 no terceiro quadrado por um 0, deixaria a cabeça de leitura/escrita onde está e passaria ao estado q_2. Se então o

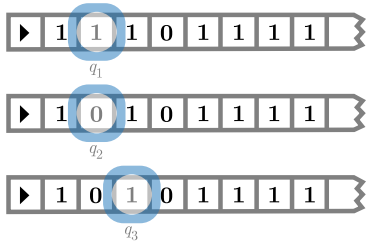


Figura 14.1: Uma máquina de Turing executando seu programa.

programa retornar $\langle q_3, 0, R\rangle$ para a entrada $\langle q_2, 0\rangle$, a máquina sobrescreveria o 0 com outro 0 (efetivamente, deixando inalterado o conteúdo da fita sob a cabeça de leitura/escrita), moveria um quadrado à direita e entraria no estado q_3. E assim por diante.

Dizemos que a máquina *para* quando encontra algum estado, q_n, e símbolo, σ, tais que não há instrução para $\langle q_n, \sigma\rangle$, ou seja, a função de transição para a entrada $\langle q_n, \sigma\rangle$ é indefinida. Em outras palavras, a máquina não tem instrução a executar e, nesse ponto, cessa a operação. A parada às vezes é representada por um estado de parada específico h. Isso será demonstrado com mais detalhes adiante.

A beleza do artigo de Turing, "On computable numbers," está em que ele apresenta não só uma definição formal, mas também um argumento de que a definição captura a noção intuitiva de computabilidade. Pela definição, deve ficar claro que toda função computável por uma máquina de Turing é computável no sentido intuitivo. Turing oferece três tipos de argumento de que o inverso é verdadeiro, ou seja, que toda função que naturalmente consideraríamos computável é computável por tal máquina. São eles (nas palavras de Turing):

1. Um apelo direto à intuição.

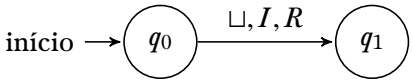
2. Uma prova da equivalência de duas definições (caso a nova definição tenha maior apelo intuitivo).

3. Dar exemplos de grandes classes de números que são computáveis.

Nosso objetivo é tentar definir a noção de computabilidade "em princípio," ou seja, sem levar em conta limitações práticas de tempo e espaço. Claro, com a definição mais ampla de computabilidade em vigor, pode-se então considerar computação com recursos limitados; isso forma o cerne do assunto conhecido como "complexidade computacional."

14.2 Representando Máquinas de Turing

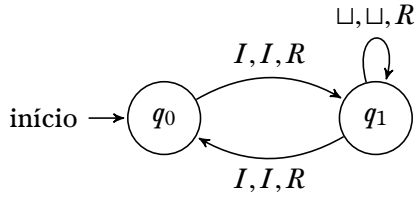
As máquinas de Turing podem ser representadas visualmente por *diagramas de estado*. Os diagramas são compostos por células de estado conectadas por setas. Sem surpresa, cada célula de estado representa um estado da máquina. Cada seta representa uma instrução que pode ser executada a partir desse estado, com as especificações da instrução escritas acima ou abaixo da seta apropriada. Considere a seguinte máquina, que tem apenas dois estados internos, q_0 e q_1, e uma instrução:



Lembre-se de que a máquina de Turing tem uma cabeça de leitura/escrita e uma fita com a entrada escrita nela. A instrução pode ser lida como *se lendo um* $\sqcup$ *no estado* q_0, *escreva um* I, *mova-se à direita, e vá para o estado* q_1. Isto é equivalente à função de transição que mapeia $\langle q_0, \sqcup \rangle$ em $\langle q_1, I, R \rangle$.

Exemplo 14.1. *Máquina Par*: A seguinte máquina de Turing para se, e somente se, houver um número par de I na fita (sob a

suposição de que todos os I vêm antes do primeiro $\sqcup$ na fita).



O diagrama de estado corresponde à seguinte função de transição:

$$\delta(q_0, I) = \langle q_1, I, R\rangle,$$
$$\delta(q_1, I) = \langle q_0, I, R\rangle,$$
$$\delta(q_1, \sqcup) = \langle q_1, \sqcup, R\rangle$$

A máquina acima para apenas quando a entrada é um número par de traços. Caso contrário, a máquina (teoricamente) continua a operar indefinidamente. Para qualquer máquina e entrada, é possível rastrear as *configurações* da máquina a fim de determinar a saída. Daremos uma definição formal de configurações mais adiante. Por enquanto, podemos pensar intuitivamente nas configurações como uma série de diagramas que mostram o estado da máquina em qualquer instante durante a operação. As configurações mostram o conteúdo da fita, o estado da máquina e a posição da cabeça de leitura/escrita.

Vamos rastrear as configurações da máquina par se ela for iniciada com uma entrada de quatro I. Nesse caso, esperamos que a máquina pare. Em seguida, executaremos a máquina com uma entrada de três I, na qual a máquina rodará para sempre.

A máquina inicia no estado q_0, examinando o I mais à esquerda. Podemos representar o estado inicial da máquina da seguinte forma:

$$\triangleright I_0 I I I \sqcup \ldots$$

A configuração acima é direta. Como se pode ver, a máquina inicia no estado um, examinando o I mais à esquerda. Isso é

representado por um subscrito do nome do estado no primeiro I. A instrução aplicável neste ponto é $\delta(q_0, I) = \langle q_1, I, R\rangle$, e assim a máquina move-se à direita na fita e muda para o estado q_1.

$$\triangleright I I_1 I I \sqcup \ldots$$

Como a máquina está agora no estado q_1 examinando um I, temos que "seguir" a instrução $\delta(q_1, I) = \langle q_0, I, R\rangle$. Isso resulta na configuração

$$\triangleright I I I_0 I \sqcup \ldots$$

À medida que a máquina continua, as regras são aplicadas novamente na mesma ordem, resultando nas duas configurações a seguir:

$$\triangleright I I I I_1 \sqcup \ldots$$

$$\triangleright I I I I \sqcup_0 \ldots$$

A máquina está agora no estado q_0 examinando um $\sqcup$. Com base no diagrama de transição, podemos ver facilmente que não há nenhuma instrução a ser executada, e portanto a máquina parou. Isso significa que a entrada foi aceita.

Suponha que, em seguida, iniciemos a máquina com uma entrada de três I. As primeiras configurações são semelhantes, pois as mesmas instruções são executadas, com apenas uma pequena diferença na entrada da fita:

$$\triangleright I_0 I I \sqcup \ldots$$

$$\triangleright I I_1 I \sqcup \ldots$$

$$\triangleright I I I_0 \sqcup \ldots$$

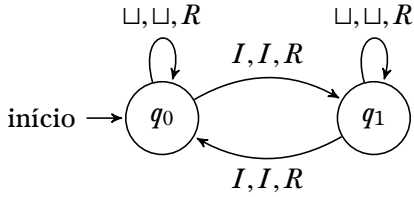
$$\triangleright I I I \sqcup_1 \ldots$$

A máquina já percorreu todos os I e está lendo um $\sqcup$ no estado q_1. Como mostrado no diagrama, há uma instrução da forma $\delta(q_1, \sqcup) = \langle q_1, \sqcup, R\rangle$. Como a fita está preenchida com $\sqcup$ indefinidamente para a direita, a máquina continuará a executar esta instrução *para sempre*, permanecendo no estado q_1 e

movendo-se cada vez mais à direita. A máquina nunca para, e não aceita a entrada.

É importante observar que nem todas as máquinas param. Se parar significa que a máquina fica sem instruções a executar, então podemos criar uma máquina que nunca para, simplesmente garantindo que haja uma seta de saída para cada símbolo em cada estado. A máquina par pode ser modificada para rodar indefinidamente adicionando uma instrução para examinar um ⊔ em q_0.

Exemplo 14.2.



As tabelas de máquina são outra forma de representar máquinas de Turing. As tabelas de máquina têm o alfabeto da fita exibido no eixo x, e o conjunto de estados da máquina ao longo do eixo y. Dentro da tabela, na interseção de cada estado e símbolo, está escrito o resto da instrução—o novo estado, o novo símbolo e a direção do movimento. As tabelas de máquina facilitam determinar em que estado, e para qual símbolo, a máquina para. Sempre que há uma lacuna na tabela, há um ponto possível para a máquina parar. Ao contrário dos diagramas de estado e dos conjuntos de instruções, em que os pontos nos quais a máquina para nem sempre são imediatamente óbvios, quaisquer pontos de parada são rapidamente identificados ao encontrar as lacunas na tabela de máquina.

Exemplo 14.3. A tabela de máquina da máquina par é:

	⊔	I	▷
q_0		I, q_1, R	
q_1	$⊔, q_1, R$	I, q_0, R	

Como podemos ver, a máquina para ao examinar um ⊔ no estado q_0.

Até aqui consideramos apenas máquinas que leem e aceitam entradas. No entanto, as máquinas de Turing têm a capacidade de tanto ler quanto escrever. Um exemplo de tal máquina (embora haja muitos, muitos exemplos) é um *duplicador*. Um duplicador, quando iniciado com um bloco de n I na fita, produz como saída um bloco de $2n$ I.

Exemplo 14.4. Antes de construir uma máquina duplicadora, é importante elaborar uma *estratégia* para resolver o problema. Como a máquina (tal como a formulamos) não consegue lembrar quantos I já leu, precisamos encontrar uma maneira de rastrear todos os I na fita. Uma dessas maneiras é separar a saída da entrada com um ⊔. A máquina pode então apagar o primeiro I da entrada, percorrer o resto da entrada, deixar um ⊔, e escrever dois novos I. A máquina então voltará e encontrará o segundo I na entrada, e duplicará esse também. Para cada I de entrada, ela escreverá dois I de saída. Ao apagar a entrada conforme a máquina avança, podemos garantir que nenhum I seja perdido ou duplicado duas vezes. Quando toda a entrada é apagada, restarão $2n$ I na fita. O diagrama de estado da máquina de Turing resultante está representado na Figura 14.2.

14.3 Máquinas de Turing

A definição formal do que constitui uma máquina de Turing parece abstrata, mas é na verdade simples: apenas reúne em uma estrutura matemática toda a informação necessária para especificar o funcionamento de uma máquina de Turing. Isso inclui (1) em quais estados a máquina pode estar, (2) quais símbolos são permitidos na fita, (3) em qual estado a máquina deve começar e (4) qual é o conjunto de instruções da máquina.

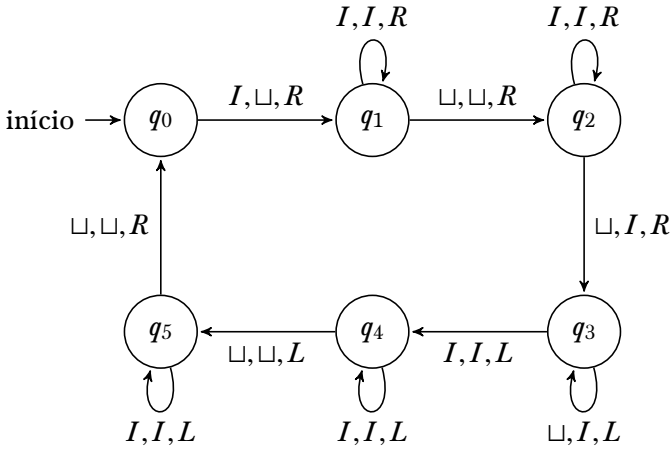


Figura 14.2: Uma máquina duplicadora

Definição 14.5 (Máquina de Turing). Uma *máquina de Turing* M é uma tupla $\langle Q, \Sigma, q_0, \delta \rangle$ consistindo de

1. um conjunto finito de *estados* Q,
2. um *alfabeto* finito Σ que inclui $\triangleright$ e $⊔$,
3. um *estado inicial* $q_0 \in Q$,
4. um *conjunto de instruções* finito $\delta \colon Q \times \Sigma \nrightarrow Q \times \Sigma \times \{L, R, N\}$.

A função parcial δ também é chamada de *função de transição* de M.

Supomos que a fita é infinita em apenas uma direção. Por essa razão é útil designar um símbolo especial $\triangleright$ como marcador da extremidade esquerda da fita. Isso facilita que programas de máquinas de Turing saibam quando estão "em perigo" de sair

da fita. Poderíamos supor que esse símbolo nunca é sobrescrito, ou seja, que $\delta(q,\triangleright) = \langle q',\triangleright,x\rangle$ se $\delta(q,\triangleright)$ estiver definido. Alguns livros-texto fazem isso; nós não. Basta ter cuidado ao construir sua máquina de Turing para que ela nunca sobrescreva $\triangleright$. Além disso, há casos em que permitir tal sobrescrita oferece alguma flexibilidade conveniente.

Exemplo 14.6. *Máquina dos pares:* A máquina dos pares é formalmente a quádrupla $\langle Q,\Sigma,q_0,\delta\rangle$ onde

$$\begin{aligned} Q &= \{q_0,q_1\} \\ \Sigma &= \{\triangleright,\sqcup,I\}, \\ \delta(q_0,I) &= \langle q_1,I,R\rangle, \\ \delta(q_1,I) &= \langle q_0,I,R\rangle, \\ \delta(q_1,\sqcup) &= \langle q_1,\sqcup,R\rangle. \end{aligned}$$

14.4 Configurações e Computações

Lembre-se de acompanhar as configurações da máquina dos pares anteriormente. O mecanismo imaginário formado pela fita, pela cabeça de leitura-escrita e pelo programa da máquina de Turing é, na verdade, apenas uma maneira intuitiva de visualizar o que é uma computação de máquina de Turing. Formalmente, podemos definir a computação de uma máquina de Turing sobre uma dada entrada como uma sequência de *configurações*—e uma configuração, por sua vez, é uma sequência de símbolos (correspondendo ao conteúdo da fita em um dado ponto da computação), um número que indica a posição da cabeça de leitura-escrita e um estado. Com isso, podemos definir o que a máquina de Turing M computa sobre uma dada entrada.

Definição 14.7 (Configuração). Uma *configuração* da máquina de Turing $M = \langle Q,\Sigma,q_0,\delta\rangle$ é uma tripla $\langle C,m,q\rangle$ onde

1. $C \in \Sigma^*$ é uma sequência finita de símbolos de Σ,

2. $m \in \mathbb{N}$ é um número $< \text{len}(C)$, e

3. $q \in Q$

Intuitivamente, a sequência C é o conteúdo da fita (símbolos de todos os quadrados desde o quadrado mais à esquerda até o último não vazio ou previamente visitado), m é o número do quadrado que a cabeça de leitura-escrita está examinando (começando com 0 como o número do quadrado mais à esquerda), e q é o estado atual da máquina.

A possível entrada para uma máquina de Turing é uma sequência de símbolos, geralmente uma sequência que codifica um número de alguma forma. A configuração inicial da máquina de Turing é aquela configuração em que iniciamos a máquina de Turing para trabalhar sobre essa entrada: a fita contém o marcador de fim de fita imediatamente seguido da entrada escrita nos quadrados à direita, a cabeça de leitura-escrita está examinando o quadrado mais à esquerda da entrada (ou seja, o quadrado à direita do marcador de fim esquerdo), e o mecanismo está no estado inicial designado q_0.

Definição 14.8 (Configuração inicial). A *configuração inicial* de M para a entrada $I \in \Sigma^*$ é

$$\langle \triangleright \frown I, 1, q_0 \rangle.$$

O símbolo $\frown$ denota *concatenação*—a cadeia de entrada começa imediatamente após o marcador de fim esquerdo.

Definição 14.9. Dizemos que uma configuração $\langle C, m, q \rangle$ *produz a configuração* $\langle C', m', q' \rangle$ *em um passo* (de acordo com M), sse

1. o m-ésimo símbolo de C é σ,

2. o conjunto de instruções de M especifica $\delta(q,\sigma) = \langle q',\sigma',D\rangle$,

3. o m-ésimo símbolo de C' é σ', e

4. a) $D = L$ e $m' = m - 1$ se $m > 0$, caso contrário $m' = 0$, ou

   b) $D = R$ e $m' = m + 1$, ou

   c) $D = N$ e $m' = m$,

5. se $m' = \text{len}(C)$, então $\text{len}(C') = \text{len}(C) + 1$ e o m'-ésimo símbolo de C' é $\sqcup$. Caso contrário $\text{len}(C') = \text{len}(C)$.

6. para todo i tal que $i < \text{len}(C)$ e $i \neq m$, $C'(i) = C(i)$,

Definição 14.10. Uma *execução de M sobre a entrada I* é uma sequência C_i de configurações de M, onde C_0 é a configuração inicial de M para a entrada I, e cada C_i produz C_{i+1} em um passo.

Dizemos que *M para sobre a entrada I após k passos* se $C_k = \langle C,m,q\rangle$, o m-ésimo símbolo de C é σ, e $\delta(q,\sigma)$ é indefinido. Nesse caso, a *saída* de M para a entrada I é O, onde O é uma cadeia de símbolos que não termina em $\sqcup$ tal que $C = \triangleright \frown O \frown \sqcup^j$ para algum $j \in \mathbb{N}$. ($\sqcup^j$ é uma sequência de j $\sqcup$'s.)

De acordo com esta definição, a saída O de M sempre termina em um símbolo diferente de $\sqcup$, ou, se no instante k toda a fita estiver preenchida com $\sqcup$ (exceto o $\triangleright$ mais à esquerda), O é a cadeia vazia.

14.5 Representação Unária de Números

As máquinas de Turing trabalham sobre sequências de símbolos escritos em sua fita. Dependendo do alfabeto que uma máquina de Turing usa, essas sequências de símbolos podem representar

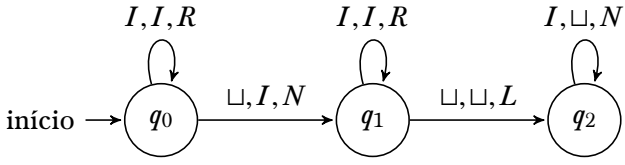


Figura 14.3: Uma máquina que computa $f(x,y) = x + y$

diversas entradas e saídas. De particular interesse, é claro, estão as máquinas de Turing que computam funções *aritméticas*, isto é, funções de números naturais. Uma maneira simples de representar inteiros positivos é codificá-los como sequências de um único símbolo I. Se $n \in \mathbb{N}$, seja I^n a sequência vazia se $n = 0$, e caso contrário a sequência composta de exatamente n I.

Definição 14.11 (Computação). Uma máquina de Turing M *computa* a função $f : \mathbb{N}^k \to \mathbb{N}$ sse M para com a entrada

$$I^{n_1} \sqcup I^{n_2} \sqcup \ldots \sqcup I^{n_k}$$

produzindo a saída $I^{f(n_1,\ldots,n_k)}$.

Exemplo 14.12. *Adição:* Vamos construir uma máquina que computa a função $f(n,m) = n + m$. Isso requer uma máquina que comece com dois blocos de I de comprimento n e m na fita, e pare com um único bloco composto de $n + m$ I. Os dois blocos de entrada de I estão separados por um $\sqcup$, então um método seria escrever um traço no quadrado que contém o $\sqcup$, e apagar o último I.

Em Exemplo 14.4, demos um exemplo de uma máquina de Turing que recebe como entrada uma sequência de I e para com uma sequência de duas vezes mais I na fita—a máquina duplicadora. No entanto, como a saída contém $\sqcup$ à esquerda do bloco

duplicado de I, ela não computa de fato a função $f(x) = 2x$, como você poderia ter suposto. Descreveremos duas maneiras de corrigir isso.

Exemplo 14.13. A máquina em Figura 14.4 computa a função $f(x) = 2x$. Em vez de apagar a entrada e escrever dois I na extremidade direita para cada I da entrada, como faz a máquina de Exemplo 14.4, esta máquina adiciona um único I à direita para cada I da entrada. Ela precisa rastrear onde a entrada termina, então deixa um $\sqcup$ entre a entrada e os traços adicionados, que ela preenche com um I bem ao final. E temos que "lembrar" onde estamos na entrada, então substituímos temporariamente um I do bloco de entrada por um $\sqcup$.

Exemplo 14.14. Uma segunda possibilidade para computar $f(x) = 2x$ é manter a máquina duplicadora original, mas adicionar estados e instruções ao final que movam o bloco duplicado de traços para a extremidade esquerda da fita. A máquina em Figura 14.5 faz exatamente esta última parte: iniciada em uma fita composta de um bloco de $\sqcup$ seguido de um bloco de I (e com a cabeça posicionada em qualquer lugar do bloco de $\sqcup$), ela apaga os I um a um e os escreve no início da fita. Para conseguir saber quando ela termina, primeiro marca o fim do bloco de I com um símbolo $\triangleright$, que é apagado ao final. Começamos a numerar os estados em q_6, de modo que possam ser adicionados à máquina duplicadora. Tudo de que você precisará é uma instrução adicional $\delta(q_0, \sqcup) = \langle q_6, \sqcup, N\rangle$, isto é, uma seta de q_0 para q_6 rotulada $\sqcup, \sqcup, N$. (Há um problema sutil: a máquina resultante não funciona para a entrada $x = 0$. Deixaremos isso como exercício.)

Definição 14.15. Uma máquina de Turing M computa a função parcial $f : \mathbb{N}^k \nrightarrow \mathbb{N}$ sse,

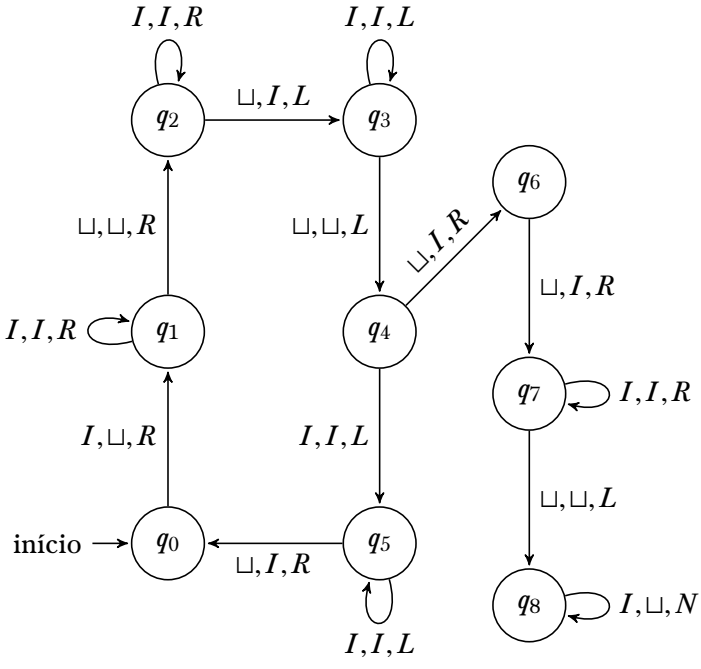


Figura 14.4: Uma máquina que computa $f(x) = 2x$

1. M para com a entrada $I^{n_1} \frown \sqcup \frown \ldots \frown \sqcup \frown I^{n_k}$ produzindo a saída I^m se $f(n_1, \ldots, n_k) = m$.

2. M não para de modo algum, ou para com uma saída que não é um único bloco de I se $f(n_1, \ldots, n_k)$ é indefinida.

14.6 Estados de Parada

Embora tenhamos definido nossas máquinas para parar apenas quando não há instrução a executar, representações comuns de

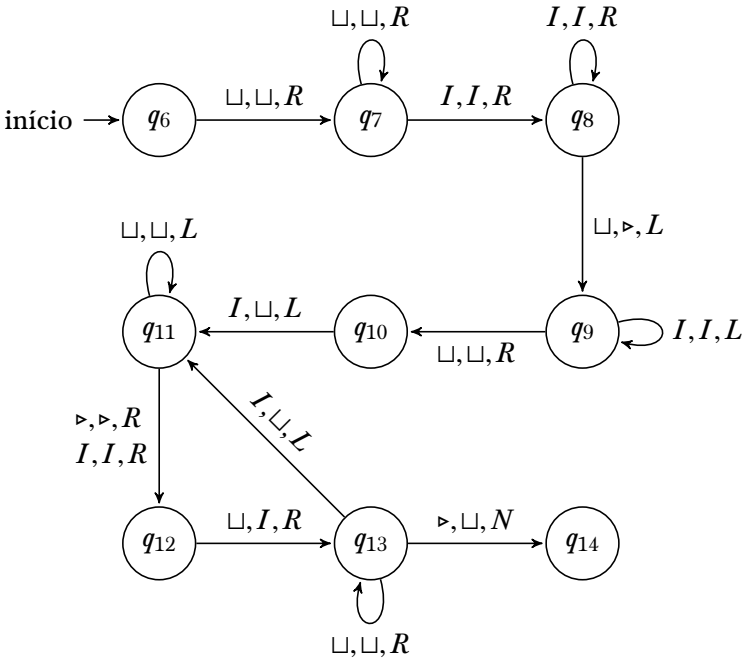


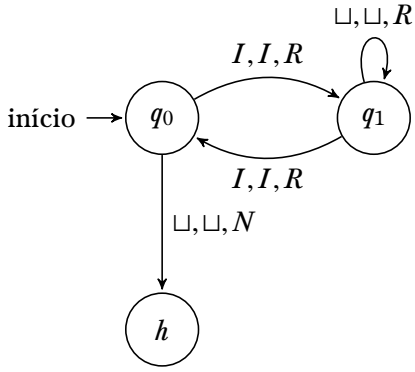
Figura 14.5: Movendo um bloco de I para a esquerda

máquinas de Turing têm um *estado de parada* dedicado h, de modo que $h \in Q$.

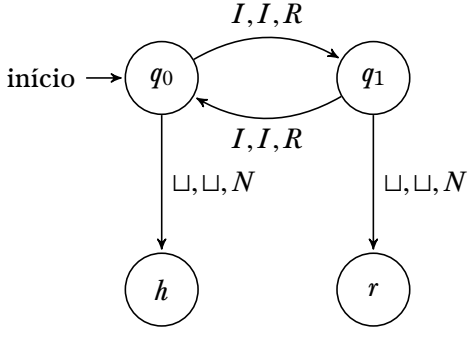
A ideia por trás de um estado de parada é simples: quando a máquina terminou a operação (está pronta para aceitar a entrada, ou terminou de escrever a saída), ela entra em um estado h onde para. Algumas máquinas têm dois estados de parada, um que aceita a entrada e outro que rejeita a entrada.

Exemplo 14.16. *Estados de parada.* Para elucidar esse conceito, comecemos com uma alteração da máquina dos pares. Em vez de fazer a máquina parar no estado q_0 se a entrada for par, podemos

adicionar uma instrução para enviar a máquina a um estado de parada.



Vamos expandir ainda mais o exemplo. Quando a máquina determina que a entrada é ímpar, ela nunca para. Podemos alterar a máquina para incluir um estado de *rejeição* substituindo a instrução em laço por uma instrução para ir a um estado de rejeição r.



Adicionar um estado de parada dedicado pode ser vantajoso em casos como este, em que deixa explícito quando a máquina aceita/rejeita certas entradas. No entanto, é importante notar que nenhum poder computacional é ganho ao adicionar um estado de parada dedicado. Da mesma forma, uma noção menos formal de parada tem suas próprias vantagens. A definição de parada

usada até agora neste capítulo torna a prova do *Problema da Parada* intuitiva e fácil de demonstrar. Por essa razão, continuamos com nossa definição original.

14.7 Máquinas Disciplinadas

Na seção 14.6, consideramos máquinas de Turing que têm um único estado de parada designado h—tais máquinas são garantidas de parar, se pararem de fato, no estado h. Dessa forma, máquinas com um único estado de parada são mais "disciplinadas" do que permitimos que máquinas de Turing sejam em geral. Há outras restrições que podemos impor ao comportamento das máquinas de Turing. Por exemplo, também não proibimos máquinas de Turing de apagar o marcador de fim de fita no quadrado 0, ou de tentar mover-se à esquerda a partir do quadrado 0. (Nossa definição afirma que a cabeça simplesmente permanece no quadrado 0 nesse caso; outras definições fazem a máquina parar.) Da mesma forma, às vezes é desejável poder supor que uma máquina de Turing, se parar de fato, para no quadrado 1.

Definição 14.17. Uma máquina de Turing M é *disciplinada* sse

1. ela tem um único estado de parada designado h,
2. ela para, se parar de fato, enquanto examina o quadrado 1,
3. ela nunca apaga o símbolo ▷ no quadrado 0, e
4. ela nunca tenta mover-se à esquerda a partir do quadrado 0.

Já discutimos que qualquer máquina de Turing pode ser transformada em uma com o mesmo comportamento, mas com um estado de parada designado. Isso é feito simplesmente adicionando um novo estado h, e adicionando uma instrução $\delta(q,\sigma) = \langle h,\sigma,N\rangle$ para qualquer par $\langle q,\sigma\rangle$ onde o δ original é indefinido. É verdade, embora tedioso de provar, que qualquer máquina de

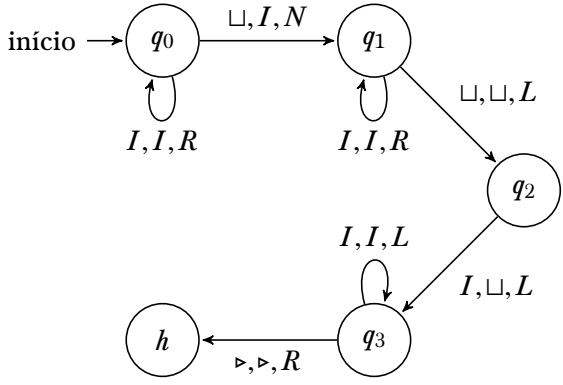


Figura 14.6: Uma máquina de adição disciplinada

Turing M pode ser transformada em uma máquina de Turing disciplinada M' que para nas mesmas entradas e produz a mesma saída. Por exemplo, se a máquina de Turing para e não está no quadrado 1, podemos adicionar algumas instruções para fazer a cabeça mover-se à esquerda até encontrar o marcador de fim de fita, depois mover um quadrado à direita, e então parar. Deixamos a você pensar sobre como as outras condições podem ser tratadas.

Exemplo 14.18. Em Figura 14.6, transformamos a máquina de adição de Exemplo 14.12 em uma máquina disciplinada.

Proposição 14.19. *Para toda máquina de Turing M, há uma máquina de Turing disciplinada M' que para com saída O se M para com saída O, e não para se M não para. Em particular, qualquer função $f : \mathbb{N}^n \to \mathbb{N}$ computável por uma máquina de Turing também é computável por uma máquina de Turing disciplinada.*

14.8 Combinando Máquinas de Turing

Os exemplos de máquinas de Turing que vimos até agora foram bastante simples por natureza. Mas, de fato, qualquer problema que possa ser resolvido com qualquer linguagem de programação moderna também pode ser resolvido com máquinas de Turing. Para construir máquinas de Turing mais complexas, é importante nos convencer de que podemos combiná-las, de modo a construir máquinas que resolvam problemas mais complexos quebrando o procedimento em partes mais simples. Se conseguirmos encontrar uma maneira natural de quebrar um problema complexo em partes constituintes, podemos atacar o problema em vários estágios, criando várias máquinas de Turing simples e combinando-as em uma única máquina capaz de resolver o problema. Esse ponto é especialmente importante ao atacar o Problema da Parada na próxima seção.

Como combinamos as máquinas de Turing $M = \langle Q, \Sigma, q_0, \delta \rangle$ e $M' = \langle Q', \Sigma', q_0', \delta' \rangle$? Usamos agora a configuração da fita após M ter parado como a configuração de entrada de uma execução da máquina M'. Para obter uma única máquina de Turing $M \frown M'$ que faz isso, faça o seguinte:

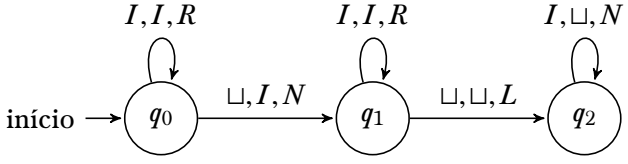
1. Renumere (ou rerrotule) todos os estados Q' de M' de modo que M e M' não tenham estados em comum ($Q \cap Q' = \emptyset$).
2. Os estados de $M \frown M'$ são $Q \cup Q'$.
3. O alfabeto da fita é $\Sigma \cup \Sigma'$.
4. O estado inicial é q_0.
5. A função de transição é a função δ'' dada por:

$$\delta''(q,\sigma) = \begin{cases} \delta(q,\sigma) & \text{se } q \in Q \\ \delta'(q,\sigma) & \text{se } q \in Q' \\ \langle q_0', \sigma, N \rangle & \text{se } q \in Q \text{ e } \delta(q,\sigma) \text{ é indefinida} \end{cases}$$

A máquina resultante usa as instruções de M quando está em um estado $q \in Q$, e as instruções de M' quando está em um estado $q \in Q'$. Quando está em um estado $q \in Q$ e examina um símbolo σ para o qual M não tem transição (isto é, M teria parado), ela entra no estado inicial de M' (e deixa o conteúdo da fita e a posição da cabeça como estão).

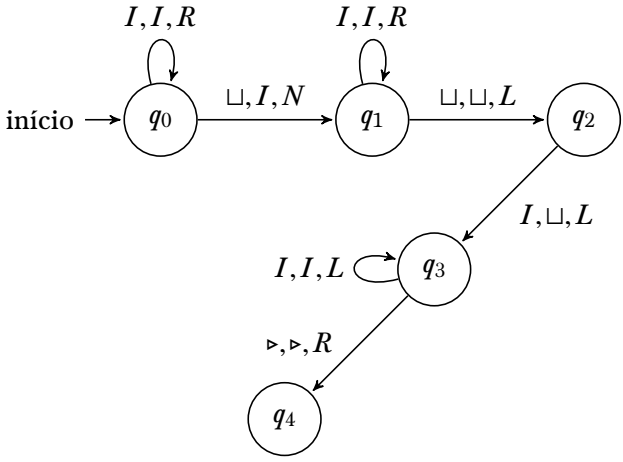
Note que, a menos que a máquina M seja disciplinada, não sabemos onde está a cabeça da fita quando M para, então a configuração de parada de M não precisa ter a cabeça examinando o quadrado 1. Ao combinar máquinas, é importante ter isso em mente.

Exemplo 14.20. *Combinando Máquinas:* Projetaremos uma máquina que, quando iniciada com uma entrada composta de dois blocos de I de comprimento n e m, para com um único bloco de $2(m+n)$ I na fita. Para construir essa máquina, podemos combinar duas máquinas que já conhecemos: a máquina de adição e a duplicadora. Começamos desenhando um diagrama de estado para a máquina de adição.



Em vez de parar no estado q_2, queremos continuar a operação a fim de duplicar a saída. Lembre-se de que a máquina duplicadora apaga o primeiro traço da entrada e escreve dois traços em uma saída separada. Vamos adicionar uma instrução para garantir que a cabeça da fita esteja lendo o primeiro traço da saída da

máquina de adição.



Agora é fácil duplicar a entrada—tudo o que temos a fazer é conectar a máquina duplicadora ao estado q_4. Isso requer renomear os estados da máquina duplicadora de modo que comecem em q_4 em vez de q_0—dessa forma não acabamos com dois estados iniciais. O diagrama final deve ter a aparência da Figura 14.7.

Proposição 14.21. *Se M e M' são disciplinadas e computam as funções $f\colon \mathbb{N}^k \to \mathbb{N}$ e $f'\colon \mathbb{N} \to \mathbb{N}$, respectivamente, então $M \frown M'$ é disciplinada e computa $f' \circ f$.*

Prova. Como M é disciplinada, quando para com saída $f(n_1, \ldots, n_k) = m$, a cabeça está examinando o quadrado 1. Se agora entrarmos no estado inicial de M', a máquina parará com saída $f'(m)$, novamente examinando o quadrado 1. As demais condições de Definição 14.17 também são satisfeitas. □

14.9 Variantes de Máquinas de Turing

Na verdade, há muitas maneiras possíveis de definir máquinas de Turing, das quais a nossa é apenas uma. De certas formas, nossa

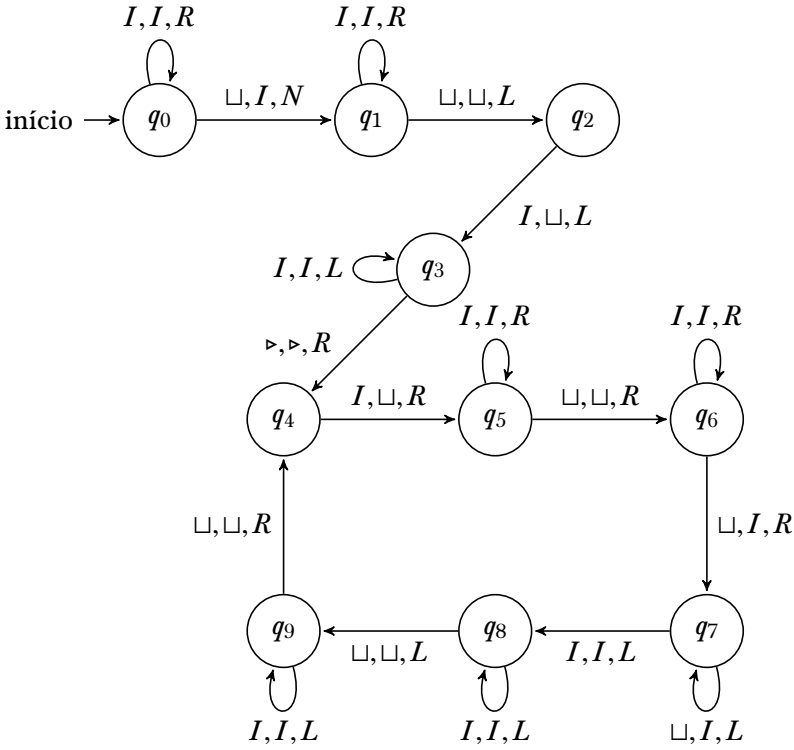


Figura 14.7: Combinando as máquinas de adição e duplicadora

definição é mais liberal do que outras. Permitimos alfabetos finitos arbitrários; uma definição mais restrita poderia permitir apenas dois símbolos de fita, I e $\sqcup$. Permitimos que a máquina escreva um símbolo na fita e se mova ao mesmo tempo; outras definições permitem escrever ou mover. Permitimos a possibilidade de escrever sem mover a cabeça da fita; outras definições omitem a "instrução" N. Em outros aspectos, nossa definição é mais restritiva. Assumimos que a fita é infinita em apenas uma direção; outras definições permitem que a fita seja infinita tanto para a esquerda quanto para a direita. Na verdade, pode-se até mesmo permitir qualquer número de fitas separadas, ou até mesmo uma grade infinita de quadrados. Representamos o conjunto de instruções da máquina de Turing por uma função de transição; outras definições usam uma relação de transição em que a máquina tem mais de uma instrução possível em qualquer situação dada.

Essa última flexibilização da definição é particularmente interessante. Em nossa definição, quando a máquina está no estado q lendo o símbolo σ, $\delta(q,\sigma)$ determina qual é o novo símbolo, o novo estado e a nova posição da cabeça da fita. Mas se permitirmos que o conjunto de instruções seja uma relação entre pares estado-símbolo atuais $\langle q,\sigma\rangle$ e novas triplas estado-símbolo-direção $\langle q',\sigma',D\rangle$, a ação da máquina de Turing pode não ser exclusivamente determinada — a relação de instrução pode conter tanto $\langle q,\sigma,q',\sigma',D\rangle$ quanto $\langle q,\sigma,q'',\sigma'',D'\rangle$. Nesse caso, temos uma máquina de Turing *não determinística.* Estas desempenham um papel importante na teoria da complexidade computacional.

Há também diferentes convenções para quando uma máquina de Turing para: dizemos que ela para quando a função de transição é indefinida; outras definições exigem que a máquina esteja em um estado de parada especial designado. Explicamos na seção 14.6 por que exigir um estado de parada designado não é uma restrição que impacta o que as máquinas de Turing podem calcular. Como as fitas de nossas máquinas de Turing são infinitas em apenas uma direção, há casos em que uma máquina de Turing não consegue executar adequadamente uma instrução: se ela lê o quadrado mais à esquerda e deveria se mover

para a esquerda. De acordo com a nossa definição, ela simplesmente permanece no lugar em vez de "cair", mas poderíamos tê-la definido de modo que ela pare quando isso acontecer. Essa definição também é equivalente: poderíamos simular o comportamento de uma máquina de Turing que para quando tenta se mover para a esquerda do quadrado 0 excluindo cada transição $\delta(q,\triangleright) = \langle q',\sigma,L\rangle$—então, em vez de tentar mover para a esquerda em $\triangleright$, a máquina para.[1]

Há também diferentes maneiras de representar números (e, portanto, a função de entrada-saída calculada por uma máquina de Turing): usamos representação unária, mas você também pode usar representação binária. Isso requer dois símbolos além de $\sqcup$ e $\triangleright$.

Agora, aqui está um fato interessante: nenhuma dessas variações importa para quais funções são Turing-computáveis. *Se uma função é Turing-computável segundo uma definição, ela é Turing-computável segundo todas elas.*

Não entraremos nos detalhes de como verificar isso. Eis apenas um exemplo: não ganhamos nenhum poder computacional adicional ao permitir uma fita que é infinita em ambas as direções, ou múltiplas fitas. A razão é, aproximadamente, que uma máquina de Turing com uma única fita infinita unidirecional pode simular múltiplas fitas ou fitas infinitas bidirecionais. Por exemplo, usando estados e instruções adicionais, podemos "traduzir" um programa para uma máquina com múltiplas fitas ou fita infinita bidirecional em um programa para uma máquina com uma única fita infinita unidirecional. A máquina traduzida pode usar os quadrados pares para os quadrados da fita 1 (ou os quadrados "positivos" de uma fita infinita bidirecional) e os quadrados ímpares para os quadrados da fita 2 (ou os quadrados "negativos").

[1] Isso não funciona *muito bem*, pois nada nos impede de escrever e ler $\triangleright$ em quadrados que não sejam o quadrado 0 (veja Exemplo 14.14). Podemos contornar isso adicionando um segundo símbolo $\triangleright'$ para usar em seu lugar para tal propósito.

14.10 A Tese de Church–Turing

As máquinas de Turing devem ser um substituto preciso do conceito de procedimento efetivo. Turing pensava que quem compreendesse tanto o conceito de procedimento efetivo quanto o de máquina de Turing teria a intuição de que tudo o que pudesse ser feito por meio de um procedimento efetivo poderia ser feito por uma máquina de Turing. Essa afirmação é sustentada pelo fato de que todas as outras propostas precisas de substituto do conceito de procedimento efetivo se mostram extensionalmente equivalentes ao conceito de máquina de Turing —ou seja, podem computar exatamente o mesmo conjunto de funções. Essa afirmação chama-se *tese de Church–Turing*.

Definição 14.22 (Tese de Church–Turing). A *tese de Church–Turing* afirma que tudo o que é computável por meio de um procedimento efetivo é computável por máquina de Turing.

A tese de Church–Turing é invocada de duas maneiras. O primeiro tipo de uso é uma desculpa para a preguiça. Suponha que temos uma descrição de um procedimento efetivo para computar algo, digamos, em "pseudocódigo." Então podemos invocar a tese de Church–Turing para justificar a afirmação de que a mesma função é computada por alguma máquina de Turing, mesmo que não a tenhamos de fato construído.

O outro uso da tese de Church–Turing é mais filosoficamente interessante. Pode-se mostrar que existem funções que não podem ser computadas por máquinas de Turing. Disso, usando a tese de Church–Turing, conclui-se que não podem ser computadas efetivamente, usando qualquer procedimento que seja. Pois se houvesse tal procedimento, pela tese de Church–Turing seguiria que haveria uma máquina de Turing para ele. Assim, se podemos provar que não há máquina de Turing que o compute, também não pode haver um procedimento efetivo. Em particular, a tese de Church–Turing é invocada para afirmar que o chamado problema da parada não só não pode ser resolvido por máquinas

de Turing, como não pode ser resolvido efetivamente de modo algum.

Resumo

Uma **máquina de Turing** é um tipo de mecanismo idealizado de computação. Ela consiste em uma **fita** infinita em um só sentido, dividida em quadrados, cada um dos quais pode conter um **símbolo** de um alfabeto predeterminado. A máquina opera movendo uma **cabeça de leitura e escrita** ao longo da fita. Ela também pode estar em um de um número predeterminado de **estados**. As ações da cabeça de leitura e escrita são determinadas por um conjunto de instruções; cada instrução está condicionada a que a máquina esteja em um certo estado e leia um certo símbolo, e especifica qual símbolo a máquina escreverá no quadrado atual, se ela moverá a cabeça de leitura e escrita um quadrado para a esquerda, para a direita, ou se permanecerá parada, e para qual estado ela mudará. Se a fita contém uma certa **entrada**, representada como uma sequência de símbolos na fita, e a máquina é colocada no estado inicial designado com a cabeça de leitura e escrita lendo o quadrado mais à esquerda da entrada, o conjunto de instruções determinará, passo a passo, uma sequência de **configurações** da máquina: conteúdo da fita, posição da cabeça de leitura e escrita e estado da máquina. Caso a máquina encontre uma configuração na qual o conjunto de instruções não contém uma instrução para a combinação atual de símbolo lido/estado, a máquina **para**, e o conteúdo da fita é a **saída**.

Números podem ser representados muito facilmente como sequências de traços na fita de uma máquina de Turing. Dizemos que uma função $\mathbb{N} \to \mathbb{N}$ é **Turing-computável** se há uma máquina de Turing que, sempre que iniciada com a representação unária de n como entrada, eventualmente para com sua fita contendo a representação unária de $f(n)$ como saída. Muitas funções aritméticas conhecidas podem ser mostradas, com facilidade (ou nem tanto), como Turing-computáveis. Muitos outros

modelos de computação além das máquinas de Turing foram propostos; e sempre se verificou que as funções aritméticas computáveis neles também são Turing-computáveis. Isso é visto como um suporte para a **Tese de Church-Turing**: a de que toda função aritmética que pode efetivamente ser computada é Turing-computável.

Problemas

Problema 14.1. Escolha uma entrada arbitrária e rastreie as configurações da máquina duplicadora em Exemplo 14.4.

Problema 14.2. Projete uma máquina de Turing com alfabeto $\{\triangleright, \sqcup, A, B\}$ que aceite, isto é, pare, qualquer cadeia de A e B na qual o número de A seja igual ao número de B *e* todos os A precedam todos os B, e que rejeite, isto é, não pare, qualquer cadeia em que o número de A não seja igual ao número de B, ou os A não precedam todos os B. (Por exemplo, a máquina deve aceitar $AABB$, e $AAABBB$, mas rejeitar tanto AAB quanto $AABBAABB$.)

Problema 14.3. Projete uma máquina de Turing com alfabeto $\{\triangleright, \sqcup, A, B\}$ que receba como entrada qualquer cadeia α de A e B e a duplique para produzir uma saída da forma $\alpha\alpha$. (Por exemplo, a entrada $ABBA$ deve resultar na saída $ABBAABBA$).

Problema 14.4. *Alfabética?:* Projete uma máquina de Turing com alfabeto $\{\triangleright, \sqcup, A, B\}$ que, ao receber como entrada uma sequência finita de A e B, verifique se todos os A aparecem à esquerda de todos os B ou não. A máquina deve deixar a cadeia de entrada na fita, e ou parar, se a cadeia for "alfabética", ou rodar para sempre, se a cadeia não for.

Problema 14.5. *Ordenador alfabético:* Projete uma máquina de Turing com alfabeto $\{\triangleright, \sqcup, A, B\}$ que receba como entrada uma sequência finita de A e B e os reorganize de modo que todos

os A fiquem à esquerda de todos os B. (Por exemplo, a sequência $BABAA$ deve tornar-se a sequência $AAABB$, e a sequência $ABBABB$ deve tornar-se a sequência $AABBBB$).

Problema 14.6. Dê uma definição para quando uma máquina de Turing M computa a função $f : \mathbb{N}^k \to \mathbb{N}^m$.

Problema 14.7. Rastreie as configurações da máquina de Exemplo 14.12 para a entrada $\langle 3, 2\rangle$. O que acontece se a máquina computa $0 + 0$?

Problema 14.8. Em Exemplo 14.14 descrevemos uma máquina composta de uma combinação da máquina duplicadora de Figura 14.4 e da máquina movedora de Figura 14.5. O que acontece se você inicia essa máquina combinada com a entrada $x = 0$, isto é, em uma fita vazia? Como você corrigiria a máquina de modo que, nesse caso, ela parasse com saída $2x = 0$? (Você deve conseguir fazer isso adicionando um estado e uma transição.)

Problema 14.9. *Subtração:* Projete uma máquina de Turing que, ao receber uma entrada de duas cadeias não vazias de traços de comprimento n e m, em que $n > m$, compute a função $f(n, m) = n - m$.

Problema 14.10. *Igualdade:* Projete uma máquina de Turing para computar a seguinte função:

$$\text{igualdade}(n, m) = \begin{cases} 1 & \text{se } n = m \\ 0 & \text{se } n \neq m \end{cases}$$

em que n e $m \in \mathbb{Z}^+$.

Problema 14.11. Projete uma máquina de Turing para computar a função $\min(x, y)$, em que x e y são inteiros positivos representados na fita por cadeias de I separadas por um $\sqcup$. Você pode usar símbolos adicionais no alfabeto da máquina.

A função min seleciona o menor valor entre seus argumentos, de modo que $\min(3,5) = 3$, $\min(20,16) = 16$, e $\min(4,4) = 4$, e assim por diante.

Problema 14.12. Dê uma máquina disciplinada que computa $f(x) = x + 1$.

Problema 14.13. Encontre uma máquina disciplinada que, quando iniciada com entrada I^n produz saída $I^n \frown \sqcup \frown I^n$.

Problema 14.14. Dê uma máquina de Turing disciplinada que computa $f(x) = x + 2$ tomando a máquina M de PROBLEMA 14.12 e construindo $M \frown M$.

CAPÍTULO 15

Indecidibilidade

15.1 Introdução

Pode parecer óbvio que nem toda função, nem mesmo toda função aritmética, pode ser computável. Há simplesmente um número grande demais delas, cujo comportamento é complicado demais. Funções definidas a partir do decaimento de partículas radioativas, por exemplo, ou de outro comportamento caótico ou aleatório. Suponha que comecemos a contar intervalos de 1 segundo a partir de um dado instante e definamos a função $f(n)$ como o número de partículas no universo que decaem no n-ésimo intervalo de 1 segundo após aquele momento inicial. Esta parece ser uma candidata a função que jamais poderemos ter esperança de computar.

Mas uma coisa é não conseguir imaginar como se computaria tais funções, e outra bem diferente é de fato provar que elas são incomputáveis. Na verdade, mesmo funções que parecem irremediavelmente complicadas podem, em um sentido abstrato, ser computáveis. Por exemplo, suponha que o universo seja finito no tempo—algum dia, num futuro muito distante, o universo se contrairá em um único ponto, como preveem algumas teorias cosmológicas. Então há apenas um número finito (mas incrivelmente grande) de segundos a partir daquele momento inicial para os quais $f(n)$ está definida. E qualquer função que esteja definida para apenas finitas entradas é computável: poderíamos listar as

saídas em uma grande tabela, ou codificá-la em um diagrama de transição de estados de máquina de Turing muito grande.

Frequentemente estamos interessados em casos especiais de funções cujos valores fornecem as respostas a perguntas de sim/não. Por exemplo, a pergunta "n é um número primo?" está associada à função

$$\text{primo}(n) = \begin{cases} 1 & \text{se } n \text{ é primo} \\ 0 & \text{caso contrário.} \end{cases}$$

Dizemos que uma pergunta de sim/não pode ser *efetivamente decidida* se a função associada de valores 1/0 é efetivamente computável.

Para provar matematicamente que há funções que não podem ser efetivamente computadas, ou problemas que não podem ser efetivamente decididos, é essencial fixar um modelo específico de computação e mostrar que há funções que ele não pode computar ou problemas que ele não pode decidir. Podemos mostrar, por exemplo, que nem toda função pode ser computada por máquinas de Turing, e nem todo problema pode ser decidido por máquinas de Turing. Podemos então apelar para a tese de Church-Turing para concluir que não apenas as máquinas de Turing não são poderosas o suficiente para computar toda função, como também nenhum procedimento efetivo o é.

A chave para provar tais resultados negativos é o fato de que podemos atribuir números às próprias máquinas de Turing. A maneira mais fácil de fazer isso é enumerá-las, talvez fixando um modo específico de escrever máquinas de Turing e seus programas e, em seguida, listando-os de maneira sistemática. Uma vez que vejamos que isso pode ser feito, então a existência de funções Turing-incomputáveis decorre de simples considerações de cardinalidade: o conjunto de funções de $\mathbb{N}$ a $\mathbb{N}$ (na verdade, mesmo apenas de $\mathbb{N}$ a $\{0,1\}$) é incontável, mas como podemos enumerar todas as máquinas de Turing, o conjunto de funções Turing-computáveis é apenas infinito contável.

Também podemos definir funções e problemas *específicos* que podemos provar ser incomputáveis e indecidíveis, respectivamente. Um desses problemas é o assim chamado *Problema da Parada.* Máquinas de Turing podem ser finitamente descritas listando-se suas instruções. Tal descrição de uma máquina de Turing, isto é, um programa de máquina de Turing, pode é claro ser usada como entrada para outra máquina de Turing. Assim, podemos considerar máquinas de Turing que decidem perguntas sobre outras máquinas de Turing. Uma pergunta particularmente interessante é esta: "A máquina de Turing dada eventualmente para quando iniciada na entrada n?" Seria bom se houvesse uma máquina de Turing que pudesse decidir essa pergunta: pense nela como uma máquina de Turing de controle de qualidade que garante que as máquinas de Turing não fiquem presas em laços infinitos e coisas do tipo. O fato interessante, que Turing provou, é que não pode haver tal máquina de Turing. Não pode haver uma única máquina de Turing que, quando iniciada em uma entrada que consiste em uma descrição de uma máquina de Turing M e algum número n, sempre pare com saída 1 ou 0 conforme a máquina M teria parado ou não quando iniciada na entrada n.

Uma vez que tenhamos exemplos de problemas indecidíveis específicos, podemos usá-los para mostrar que outros problemas também são indecidíveis. Por exemplo, um célebre problema indecidível é a pergunta "A fórmula A de primeira ordem é válida?". Não há máquina de Turing que, dada como entrada uma fórmula A de primeira ordem, tenha garantia de parar com saída 1 ou 0 conforme A seja válida ou não. Historicamente, a questão de encontrar um procedimento para resolver efetivamente esse problema foi chamada simplesmente de "o" problema da decisão; e assim dizemos que o problema da decisão é insolúvel. Turing e Church provaram esse resultado independentemente mais ou menos na mesma época, de modo que ele também é chamado de Teorema de Church-Turing.

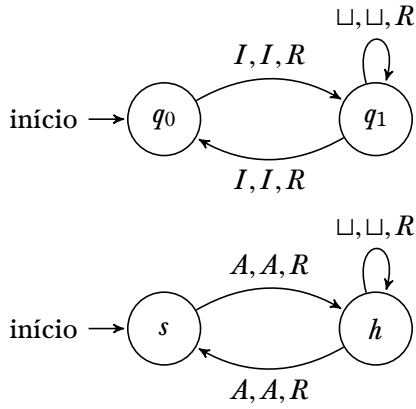


Figura 15.1: Variantes da máquina *Par*

15.2 Enumerando Máquinas de Turing

Podemos mostrar que o conjunto de todas as máquinas de Turing é contável. Isso decorre do fato de que cada máquina de Turing pode ser finitamente descrita. O conjunto de estados e o vocabulário da fita são conjuntos finitos. A função de transição é uma função parcial de $Q \times \Sigma$ a $Q \times \Sigma \times \{L, R, N\}$ e, portanto, também pode ser especificada listando-se seus valores para os finitos pares de argumentos para os quais está definida.

Isso é verdade até certo ponto, mas há uma diferença sutil. A definição de máquinas de Turing não fez restrição alguma sobre quais elementos o conjunto de estados e o alfabeto da fita podem ter. Assim, por exemplo, para todo número real, há tecnicamente uma máquina de Turing que usa esse número como um estado. Contudo, o *comportamento* da máquina de Turing é independente de quais objetos servem como estados e vocabulário. Considere as duas máquinas de Turing na Figura 15.1. Esses dois diagramas correspondem a duas máquinas, M com o alfabeto de fita $\Sigma = \{\triangleright, \sqcup, I\}$ e conjunto de estados $\{q_0, q_1\}$, e M' com alfabeto $\Sigma' = \{\triangleright, \sqcup, A\}$ e estados $\{s, h\}$. Mas, quanto ao resto, suas ins-

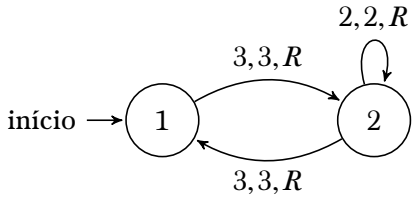


Figura 15.2: Uma máquina *Par* padrão

truções são as mesmas: M para em uma sequência de n I se, e somente se, n é par, e M' para em uma sequência de n A se, e somente se, n é par. Tudo o que fizemos foi renomear I para A, q_0 para s e q_1 para h. Esse exemplo se generaliza: podemos considerar máquinas de Turing como sendo a mesma desde que uma resulte da outra por tal renomeação de símbolos e estados. Na verdade, podemos simplesmente pensar nos símbolos e estados de uma máquina de Turing como inteiros positivos: em vez de σ_0 pense 1, em vez de σ_1 pense 2, etc.; ▷ é 1, ␣ é 2, etc. Dessa forma, a máquina *Par* torna-se a máquina representada na Figura 15.2. Poderíamos chamar de máquina *padrão* uma máquina de Turing cujos estados e símbolos são inteiros positivos, e considerar apenas máquinas padrão de agora em diante.[1]

Queríamos mostrar que o conjunto de máquinas de Turing é contável e, tendo em mente as considerações acima, basta mostrar que o conjunto de máquinas de Turing padrão é contável. Suponha que nos seja dada uma máquina de Turing padrão $M = \langle Q, \Sigma, q_0, \delta \rangle$. Como poderíamos descrevê-la usando uma cadeia finita de inteiros positivos? Primeiro listaremos o número de estados, os próprios estados, o número de símbolos, os próprios símbolos e o estado inicial. (Lembre-se, todos esses são inteiros positivos, já que M é uma máquina padrão.) E quanto

[1] A terminologia "máquina padrão" não é padrão.

a δ? O conjunto de argumentos possíveis, isto é, pares $\langle q, \sigma \rangle$, é finito, já que Q e Σ são finitos. Assim, a informação em δ é simplesmente a lista finita de todas as 5-uplas $\langle q, \sigma, q', \sigma', d \rangle$ em que $\delta(q, \sigma) = \langle q', \sigma', D \rangle$, e d é um número que codifica a direção D (digamos, 1 para L, 2 para R e 3 para N).

Dessa forma, toda máquina de Turing padrão pode ser descrita por uma lista finita de inteiros positivos, isto é, como uma sequência $s_M \in (\mathbb{Z}^+)^*$. Por exemplo, a máquina *Par* padrão é codificada pela sequência

$$2, \underbrace{1,2}_{Q}, 3, \overbrace{1,2,3}^{\Sigma}, 1, \underbrace{1,3,2,3,2}_{\delta(1,3)=\langle 2,3,R\rangle}, \overbrace{2,2,2,2,2}^{\delta(2,2)=\langle 2,2,R\rangle}, \underbrace{2,3,1,3,2}_{\delta(2,3)=\langle 1,3,R\rangle}.$$

Teorema 15.1. *Há funções de $\mathbb{N}$ a $\mathbb{N}$ que não são Turing-computáveis.*

Prova. Sabemos que o conjunto de sequências finitas de inteiros positivos $(\mathbb{Z}^+)^*$ é contável (PROBLEMA 4.7). Isso nos dá que o conjunto de descrições de máquinas de Turing padrão, como um subconjunto de $(\mathbb{Z}^+)^*$, é ele próprio enumerável. Toda função Turing-computável de $\mathbb{N}$ a $\mathbb{N}$ é computada por alguma (na verdade, muitas) máquina de Turing. Renomeando seus estados e símbolos para inteiros positivos (em particular, $\triangleright$ como 1, $\sqcup$ como 2 e I como 3), podemos ver que toda função Turing-computável é computada por uma máquina de Turing padrão. Isso significa que o conjunto de todas as funções Turing-computáveis de $\mathbb{N}$ a $\mathbb{N}$ é também enumerável.

Por outro lado, o conjunto de todas as funções de $\mathbb{N}$ a $\mathbb{N}$ não é contável (PROBLEMA 4.21). Se todas as funções fossem computáveis por alguma máquina de Turing, poderíamos enumerar o conjunto de todas as funções listando todas as descrições de máquinas de Turing que as computam. Logo, há algumas funções que não são Turing-computáveis. □

15.3 Máquinas de Turing Universais

Na seção 15.2 discutimos como toda máquina de Turing pode ser descrita por uma sequência finita de inteiros. Essa sequência codifica os estados, o alfabeto, o estado inicial e as instruções da máquina de Turing. Também observamos que o conjunto de todas essas descrições é contável. Como o conjunto de tais descrições é infinito contável, isso significa que há uma função sobrejetora de $\mathbb{N}$ nessas descrições. Uma tal função sobrejetora pode ser obtida, por exemplo, usando o método zigue-zague de Cantor. Ele nos dá uma maneira de enumerar todas as (descrições de) máquinas de Turing. Se fixarmos uma tal enumeração, passa a fazer sentido falar da primeira, da segunda, ..., e-ésima máquina de Turing. Esses números são chamados de *índices*.

Definição 15.2. Se M é a e-ésima máquina de Turing (em nossa enumeração fixada), dizemos que e é um *índice* de M. Escrevemos M_e para a e-ésima máquina de Turing.

Uma máquina pode ter mais de um índice; por exemplo, duas descrições de M podem diferir na ordem em que listamos suas instruções, e essas diferentes descrições terão índices diferentes.

É importante notar que é possível dar a enumeração das descrições de máquinas de Turing de tal modo que possamos computar efetivamente a descrição de M a partir de seu índice, e computar efetivamente um índice de uma máquina M a partir de sua descrição. Pela tese de Church-Turing, é então possível encontrar uma máquina de Turing que recupera a descrição da máquina de Turing com índice e e escreve a descrição correspondente em sua fita como saída. A descrição seria uma sequência de blocos de I (representando os inteiros positivos na sequência que descreve M_e).

Diante disso, torna-se natural perguntar: que funções de índices de máquinas de Turing são elas próprias computáveis por máquinas de Turing? Que propriedades de índices de máquinas de Turing podem ser decididas por máquinas de Turing? Um

exemplo: a função que leva um índice e ao número de estados que a máquina de Turing com índice e possui é computável por uma máquina de Turing. Eis o que tal máquina de Turing faria: iniciada em uma fita contendo um único bloco de e I, ela primeiro decodificaria e em sua descrição. A descrição passa a ser representada por uma sequência de blocos de I na fita. Como o primeiro elemento nessa sequência é o número de estados, tudo o que resta a fazer agora é apagar tudo, exceto o primeiro bloco de I, e então parar.

Um resultado notável é o seguinte:

Teorema 15.3. *Há uma* máquina de Turing universal U *que, quando iniciada na entrada* $\langle e, n\rangle$,

1. *para se, e somente se,* M_e *para na entrada* n, *e*
2. *se* M_e *para com saída* m, *então* U *também para.*

Assim, U *computa a função* $f : \mathbb{N}\times\mathbb{N} \nrightarrow \mathbb{N}$ *dada por* $f(e, n) = m$ *se* M_e *iniciada na entrada* n *para com saída* m, *e indefinida caso contrário.*

Prova. Produzir de fato U é basicamente impossível, já que se trata de uma máquina extremamente complicada. Mas podemos descrever em linhas gerais como ela funciona, e então invocar a tese de Church-Turing. Quando ela começa, a fita de U contém um bloco de e I seguido de um bloco de n I. Ela primeiro "decodifica" o índice e à direita da entrada n. Isso produz uma lista de números (isto é, blocos de I separados por ⊔) que descreve as instruções da máquina M_e. Em seguida, U escreve o número do estado inicial de M_e e o número 1 na fita, à direita da descrição de M_e. (Novamente, esses são representados em unário, como blocos de I.) Depois, ela copia a entrada (bloco de n I) para a direita—mas substitui cada I por um bloco de três I (lembre-se, o número do símbolo I é 3, sendo 1 o número de ▷ e 2 o número de ⊔). Na extremidade esquerda dessa sequência de blocos (separados por símbolos ⊔ na fita de U), ela escreve um único I, o código de ▷.

U agora tem em sua fita: o índice e, o número n, o número de código do estado inicial (o "estado atual"), o número da posição inicial do cabeçote 1 (a "posição atual do cabeçote") e o conteúdo inicial da "fita" (uma sequência de blocos de I representando os números de código dos símbolos de M_e—os "símbolos"—separados por ␣).

Ela agora simula o que M_e faria se iniciada na entrada n, fazendo o seguinte:

1. Encontrar o número k da "posição atual do cabeçote" (no início, esse é 1),
2. Mover-se para o k-ésimo bloco na "fita" para ver qual é o "símbolo" ali,
3. Encontrar a instrução que corresponde ao "estado" e ao "símbolo" atuais,
4. Voltar para o k-ésimo bloco na "fita" e substituir o "símbolo" ali pelo número de código do símbolo que M_e escreveria,
5. Mover o cabeçote para onde ela registra o "estado" atual e substituir o número ali pelo número do novo estado,
6. Mover-se para o lugar onde ela registra a "posição na fita" e apagar um I ou adicionar um I (se a instrução mandar mover para a esquerda ou para a direita, respectivamente).
7. Repetir.[2]

Se M_e iniciada na entrada n nunca para, então U também nunca para, de modo que sua saída é indefinida.

Se no passo (3) acontecer de a descrição de M_e não conter nenhuma instrução para o par "estado"/"símbolo" atual, então

[2]Estamos passando por cima de algumas dificuldades sutis aqui. Por exemplo, U pode precisar de algum espaço extra quando incrementa o contador onde mantém o registro da "posição atual do cabeçote"—nesse caso, terá de mover toda a "fita" para a direita.

M_e pararia. Se isso acontecer, U apaga a parte de sua fita à esquerda da "fita". Para cada bloco de três I (representando um I na fita de M_e), ela escreve um I na extremidade esquerda de sua própria fita, e apaga sucessivamente a "fita". Quando isso está concluído, a fita de U contém um único bloco de I de comprimento m.

Se U encontrar algo diferente de um bloco de três I na "fita", ela para imediatamente. Como a fita de U nesse caso não contém um único bloco de I, sua saída não é um número natural, isto é, $f(e, n)$ é indefinida nesse caso. □

15.4 O Problema da Parada

Suponha que tenhamos fixado alguma enumeração das descrições de máquinas de Turing. Cada máquina de Turing recebe assim um *índice*: seu lugar na enumeração M_1, M_2, M_3, ... das descrições de máquinas de Turing.

Sabemos que devem existir funções não-Turing-computáveis: o conjunto de descrições de máquinas de Turing—e, portanto, o conjunto de máquinas de Turing—é contável, mas o conjunto de todas as funções de $\mathbb{N}$ a $\mathbb{N}$ não é. Mas também podemos encontrar exemplos específicos de funções não computáveis. Uma dessas funções é a função de parada.

Definição 15.4 (Função de parada). A *função de parada* h é definida como

$$h(e, n) = \begin{cases} 0 & \text{se a máquina } M_e \text{ não para na entrada } n \\ 1 & \text{se a máquina } M_e \text{ para na entrada } n \end{cases}$$

Definição 15.5 (Problema da parada). O *Problema da Parada* é o problema de determinar (para quaisquer e, n) se a máquina de Turing M_e para com uma entrada de n traços.

Mostramos que h não é Turing-computável mostrando que uma função relacionada, s, não é Turing-computável. Esta prova baseia-se no fato de que tudo o que pode ser computado por uma máquina de Turing pode ser computado por uma máquina de Turing disciplinada (seção 14.7), e no fato de que duas máquinas de Turing podem ser conectadas para criar uma única máquina (seção 14.8).

Definição 15.6. A função s é definida como

$$s(e) = \begin{cases} 0 & \text{se a máquina } M_e \text{ não para na entrada } e \\ 1 & \text{se a máquina } M_e \text{ para na entrada } e \end{cases}$$

Lema 15.7. *A função s não é Turing-computável.*

Prova. Suponha, por contradição, que a função s seja Turing-computável. Então haveria uma máquina de Turing S que computa s. Podemos supor, sem perda de generalidade, que quando S para, ela o faz examinando o primeiro quadrado (isto é, que ela é disciplinada). Esta máquina pode ser "conectada" a outra máquina J, que para se for iniciada na entrada 0 (isto é, se ela lê $\sqcup$ no estado inicial enquanto examina o quadrado à direita do símbolo de fim de fita), e caso contrário vagueia para a direita, sem nunca parar. $S \frown J$, a máquina criada conectando-se S a J, é uma máquina de Turing, logo é M_e para algum e (isto é, aparece em algum lugar da enumeração). Inicie M_e em uma entrada de e I. Há duas possibilidades: ou M_e para ou não para.

1. Suponha que M_e pare para uma entrada de e I. Então $s(e) = 1$. Assim, S, quando iniciada em e, para com um único I como saída na fita. Então J começa com um I na fita. Nesse caso, J não para. Mas M_e é a máquina $S \frown J$, de modo que ela deveria fazer exatamente o que S seguida de J faria (isto é, neste caso, vaguear para a direita e nunca parar). Logo, M_e não pode parar para uma entrada de e I.

2. Agora suponha que M_e não pare para uma entrada de e I. Então $s(e) = 0$, e S, quando iniciada na entrada e, para com uma fita em branco. J, quando iniciada em uma fita em branco, para imediatamente. Novamente, M_e faz o que S seguida de J faria, de modo que M_e deve parar para uma entrada de e I.

Em cada caso chegamos a uma contradição com nossa suposição. Isso mostra que não pode haver uma máquina de Turing S: s não é Turing-computável. □

Teorema 15.8 (Insolubilidade do Problema da Parada). *O problema da parada é insolúvel, isto é, a função h não é Turing-computável.*

Prova. Suponha que h fosse Turing-computável, digamos, por uma máquina de Turing H. Poderíamos usar H para construir uma máquina de Turing que computa s: primeiro, fazer uma cópia da entrada (separada por um símbolo ␣). Depois, voltar ao início e executar H. Claramente podemos fazer uma máquina que realiza a primeira tarefa (veja PROBLEMA 14.13), e se H existisse, poderíamos "conectá-la" a tal máquina copiadora para obter uma nova máquina que determinaria se M_e para na entrada e, isto é, computa s. Mas já mostramos que não pode existir tal máquina. Portanto, h também não é Turing-computável. □

15.5 O Problema da Decisão

Dizemos que a lógica de primeira ordem é *decidível* se, e somente se, há um método efetivo para determinar se uma dada sentença é válida ou não. Acontece que não existe tal método: o problema de decidir a validade de sentenças de primeira ordem é insolúvel.

A fim de estabelecer esse importante resultado negativo, provamos que o problema da decisão não pode ser resolvido por

uma máquina de Turing. Isto é, mostramos que não existe máquina de Turing que, sempre que iniciada em uma fita que contém uma sentença de primeira ordem, eventualmente para e produz 1 ou 0 conforme a sentença seja válida ou não. Pela tese de Church-Turing, toda função que é computável é Turing-computável. Logo, se essa "função de validade" fosse de algum modo efetivamente computável, ela seria Turing-computável. Se ela não é Turing-computável, então também não pode ser efetivamente computável.

Nossa estratégia para provar que o problema da decisão é insolúvel é reduzir o problema da parada a ele. Isso significa o seguinte: provamos que a função $h(e,w)$ que para com saída 1 se a máquina de Turing descrita por e para na entrada w e produz 0 caso contrário não é Turing-computável. Mostraremos que se houvesse uma máquina de Turing que decide a validade de sentenças de primeira ordem, então haveria também uma máquina de Turing que computa h. Como h não pode ser computada por uma máquina de Turing, também não pode haver uma máquina de Turing que decide a validade.

O primeiro passo nessa estratégia é mostrar que, para toda entrada w e máquina de Turing M, podemos descrever efetivamente uma sentença $T(M,w)$ representando o conjunto de instruções de M e a entrada w, e uma sentença $E(M,w)$ exprimindo "M eventualmente para", tal que:

> $\models T(M,w) \rightarrow E(M,w)$ se, e somente se, M para na entrada w.

A maior parte de nossa prova consistirá em descrever essas sentenças $T(M,w)$ e $E(M,w)$ e em verificar que $T(M,w) \rightarrow E(M,w)$ é válida se, e somente se, M para na entrada w.

15.6 Representando Máquinas de Turing

A fim de representar máquinas de Turing e seu comportamento por meio de uma sentença da lógica de primeira ordem, temos de

definir uma linguagem adequada. A linguagem consiste em duas partes: predicados para descrever configurações da máquina, e expressões para numerar os passos de execução ("momentos") e as posições na fita.

Introduzimos dois tipos de predicados, ambos de 2 lugares: para cada estado q, um predicado Q_q, e para cada símbolo de fita σ, um predicado S_σ. Os primeiros permitem-nos descrever o estado de M e a posição de seu cabeçote de fita; os segundos permitem-nos descrever o conteúdo da fita.

A fim de exprimir as posições do cabeçote de fita e o número de passos executados, precisamos de uma maneira de exprimir números. Isso é feito usando um símbolo de constante o e uma função de 1 lugar $\prime$, a função sucessor. Por convenção, ela é escrita *depois* de seu argumento (e omitimos os parênteses).

Para cada número n há um termo canônico $\overline{n}$, o *numeral* de n, que o representa em $\mathscr{L}_M$. $\overline{0}$ é o, $\overline{1}$ é o', $\overline{2}$ é o'', e assim por diante. Mais formalmente:

$$\overline{0} = \mathsf{o}$$
$$\overline{n+1} = \overline{n}'$$

O termo $\overline{0}$, isto é, o, nomeia a posição mais à esquerda na fita, bem como o instante anterior ao primeiro passo de execução (a configuração inicial). O termo $\overline{1}$, isto é, o', nomeia o quadrado à direita do quadrado mais à esquerda, e o instante posterior ao primeiro passo de execução, e assim por diante.

Também introduzimos um predicado $<$ para exprimir tanto a ordenação das posições na fita (quando significa "à esquerda de") quanto a dos passos de execução (caso em que significa "antes").

Uma vez que tenhamos a linguagem montada, listamos os "axiomas" de $T(M,w)$, isto é, as sentenças que, tomadas em conjunto, descrevem o comportamento de M quando executada na entrada w. Haverá sentenças que estabelecem condições sobre o, $\prime$ e $<$, sentenças que descrevem a configuração de entrada, e sentenças que descrevem qual é a configuração de M após executar uma instrução particular.

Definição 15.9. Dada uma máquina de Turing $M = \langle Q, \Sigma, q_0, \delta \rangle$, a linguagem $\mathcal{L}_M$ consiste em:

1. Um predicado de dois lugares $Q_q(x, y)$ para todo estado $q \in Q$. Intuitivamente, $Q_q(\overline{m}, \overline{n})$ exprime "após n passos, M está no estado q examinando o m-ésimo quadrado."
2. Um predicado de dois lugares $S_\sigma(x, y)$ para todo símbolo $\sigma \in \Sigma$. Intuitivamente, $S_\sigma(\overline{m}, \overline{n})$ exprime "após n passos, o m-ésimo quadrado contém o símbolo σ."
3. Um símbolo de constante 0
4. Um símbolo de função de um lugar $'$
5. Um predicado de dois lugares $<$

As sentenças que descrevem a operação da máquina de Turing M na entrada $w = \sigma_{i_1} \ldots \sigma_{i_k}$ são as seguintes:

1. Axiomas que descrevem números e $<$:
   a) Uma sentença que diz que todo número é menor que seu sucessor:
   $$\forall x \; x < x'$$
   b) Uma sentença que garante que $<$ é transitivo:
   $$\forall x \, \forall y \, \forall z \, ((x < y \wedge y < z) \rightarrow x < z)$$
2. Axiomas que descrevem a configuração de entrada:
   a) Após 0 passos—antes de a máquina começar—M está no estado inicial q_0, examinando o quadrado 1:
   $$Q_{q_0}(\overline{1}, \overline{0})$$
   b) Os primeiros $k + 1$ quadrados contêm os símbolos $\triangleright$, $\sigma_{i_1}, \ldots, \sigma_{i_k}$:
   $$S_\triangleright(\overline{0}, \overline{0}) \wedge S_{\sigma_{i_1}}(\overline{1}, \overline{0}) \wedge \cdots \wedge S_{\sigma_{i_k}}(\overline{k}, \overline{0})$$

c) Caso contrário, a fita está vazia:

$$\forall x\,(\overline{k} < x \rightarrow S_{\sqcup}(x,\overline{0}))$$

3. Axiomas que descrevem a transição de uma configuração para a seguinte:

   Para o que segue, seja $A(x,y)$ a conjunção de todas as sentenças da forma

$$\forall z\,(((z < x \vee x < z) \wedge S_{\sigma}(z,y)) \rightarrow S_{\sigma}(z,y'))$$

   em que $\sigma \in \Sigma$. Usamos $A(\overline{m},\overline{n})$ para exprimir "exceto no quadrado m, a fita após $n+1$ passos é a mesma que após n passos."

   a) Para toda instrução $\delta(q_i,\sigma) = \langle q_j,\sigma',R\rangle$, a sentença:

$$\begin{aligned}\forall x\,\forall y\,((&Q_{q_i}(x,y) \wedge S_{\sigma}(x,y)) \rightarrow \\ &(Q_{q_j}(x',y') \wedge S_{\sigma'}(x,y') \wedge A(x,y)))\end{aligned}$$

   Isto diz que se, após y passos, a máquina está no estado q_i examinando o quadrado x que contém o símbolo σ, então após $y+1$ passos ela está examinando o quadrado $x+1$, está no estado q_j, o quadrado x agora contém σ', e todo quadrado distinto de x contém o mesmo símbolo que continha após y passos.

   b) Para toda instrução $\delta(q_i,\sigma) = \langle q_j,\sigma',L\rangle$, a sentença:

$$\begin{aligned}\forall x\,\forall y\,((&Q_{q_i}(x',y) \wedge S_{\sigma}(x',y)) \rightarrow \\ &(Q_{q_j}(x,y') \wedge S_{\sigma'}(x',y') \wedge A(x,y))) \wedge \\ \forall y\,((&Q_{q_i}(\overline{0},y) \wedge S_{\sigma}(\overline{0},y)) \rightarrow \\ &(Q_{q_j}(\overline{0},y') \wedge S_{\sigma'}(\overline{0},y') \wedge A(\overline{0},y)))\end{aligned}$$

   Reserve um momento para pensar em como isso funciona: agora não começamos com "se examinando o quadrado x ...", mas: "se examinando o quadrado

$x + 1 \ldots$". Um movimento para a esquerda significa que no próximo passo a máquina está examinando o quadrado x. Mas o quadrado em que se escreve é x+1. Fazemos assim porque não temos subtração nem uma função predecessor.

Note que os números da forma $x + 1$ são 1, 2, ..., isto é, isso não cobre o caso em que a máquina está examinando o quadrado 0 e deveria mover-se para a esquerda (o que, é claro, não pode fazer—ela simplesmente permanece onde está). Esse caso especial é coberto pela segunda conjunção: ela diz que se, após y passos, a máquina está examinando o quadrado 0 no estado q_i e o quadrado 0 contém o símbolo σ, então após $y + 1$ passos ela ainda está examinando o quadrado 0, está agora no estado q_j, o símbolo no quadrado 0 é σ', e os quadrados distintos do quadrado 0 contêm os mesmos símbolos que continham após y passos.

c) Para toda instrução $\delta(q_i,\sigma) = \langle q_j,\sigma',N\rangle$, a sentença:

$$\forall x\,\forall y\,((Q_{q_i}(x,y) \wedge S_\sigma(x,y)) \rightarrow (Q_{q_j}(x,y') \wedge S_{\sigma'}(x,y') \wedge A(x,y)))$$

Seja $T(M,w)$ a conjunção de todas as sentenças acima para a máquina de Turing M e a entrada w.

A fim de exprimir que M eventualmente para, temos de encontrar uma sentença que diga "após algum número de passos, a função de transição será indefinida." Seja X o conjunto de todos os pares $\langle q,\sigma\rangle$ tais que $\delta(q,\sigma)$ é indefinida. Seja então $E(M,w)$ a sentença

$$\exists x\,\exists y\,(\bigvee_{\langle q,\sigma\rangle\in X}(Q_q(x,y) \wedge S_\sigma(x,y)))$$

Se usarmos uma máquina de Turing com um estado de parada designado h, é ainda mais fácil: então a sentença $E(M,w)$

$$\exists x\,\exists y\,Q_h(x,y)$$

exprime que a máquina eventualmente para.

Proposição 15.10. *Se $m < k$, então $T(M,w) \vDash \overline{m} < \overline{k}$*

Prova. Exercício. □

15.7 Verificando a Representação

A fim de verificar que nossa representação funciona, temos de provar duas coisas. Primeiro, temos de mostrar que se M para na entrada w, então $T(M,w) \rightarrow E(M,w)$ é válida. Em seguida, temos de mostrar a recíproca, isto é, que se $T(M,w) \rightarrow E(M,w)$ é válida, então M de fato eventualmente para quando executada na entrada w.

A estratégia para prová-las é bem diferente. Para o primeiro resultado, temos de mostrar que uma sentença da lógica de primeira ordem (a saber, $T(M,w) \rightarrow E(M,w)$) é válida. A maneira mais fácil de fazer isso é dar uma derivação. Nossa prova, porém, deve funcionar para todos os M e w, de modo que não há realmente uma única sentença para a qual tenhamos de dar uma derivação, mas infinitas. Assim, o melhor que podemos fazer é provar por indução que, sejam quais forem M e w, e por mais passos que M leve para parar na entrada w, haverá uma derivação de $T(M,w) \rightarrow E(M,w)$.

Naturalmente, nossa indução procederá sobre o número de passos que M leva antes de atingir uma configuração de parada. Em nossa prova indutiva, estabeleceremos que para cada passo n da execução de M na entrada w, $T(M,w) \vDash C(M,w,n)$, em que $C(M,w,n)$ descreve corretamente a configuração de M executada em w após n passos. Ora, se M para na entrada w após, digamos, n passos, $C(M,w,n)$ descreverá uma configuração de parada. Mostraremos também que $C(M,w,n) \vDash E(M,w)$ sempre que $C(M,w,n)$ descreve uma configuração de parada. Assim, se M para na entrada w, então para algum n, M estará em uma configuração de parada após n passos. Logo,

$T(M,w) \vDash C(M,w,n)$ em que $C(M,w,n)$ descreve uma configuração de parada, e como nesse caso $C(M,w,n) \vDash E(M,w)$, obtemos que $T(M,w) \vDash E(M,w)$, isto é, que $\vDash T(M,w) \to E(M,w)$.

A estratégia para a recíproca é bem diferente. Aqui supomos que $\vDash T(M,w) \to E(M,w)$ e temos de provar que M para na entrada w. Da hipótese obtemos que $T(M,w) \vDash E(M,w)$, isto é, $E(M,w)$ é verdadeira em toda estrutura em que $T(M,w)$ é verdadeira. Assim, descreveremos uma estrutura $\boldsymbol{M}$ em que $T(M,w)$ é verdadeira: seu domínio será $\mathbb{N}$, e a interpretação de todos os Q_q e S_σ será dada pelas configurações de M durante uma execução na entrada w. Assim, por exemplo, $\boldsymbol{M} \vDash Q_q(\overline{m},\overline{n})$ se, e somente se, T, quando executada na entrada w por n passos, está no estado q e examinando o quadrado m. Ora, como $T(M,w) \vDash E(M,w)$ por hipótese, e como $\boldsymbol{M} \vDash T(M,w)$ por construção, $\boldsymbol{M} \vDash E(M,w)$. Mas $\boldsymbol{M} \vDash E(M,w)$ se, e somente se, há algum $n \in |M| = \mathbb{N}$ tal que M, executada na entrada w, está em uma configuração de parada após n passos.

Definição 15.11. Seja $C(M,w,n)$ a sentença

$$Q_q(\overline{m},\overline{n}) \wedge S_{\sigma_0}(\overline{0},\overline{n}) \wedge \cdots \wedge S_{\sigma_k}(\overline{k},\overline{n}) \wedge \forall x\,(\overline{k} < x \to S_\sqcup(x,\overline{n}))$$

em que q é o estado de M no instante n, M está examinando o quadrado m no instante n, o quadrado i contém o símbolo σ_i no instante n para $0 \leq i \leq k$, e k é o quadrado não-vazio mais à direita da fita no instante 0, ou o quadrado mais à direita que o cabeçote de fita visitou após n passos, o que for maior.

Lema 15.12. *Se M executada na entrada w está em uma configuração de parada após n passos, então $C(M,w,n) \vDash E(M,w)$.*

Prova. Suponha que M pare para a entrada w após n passos. Há algum estado q, quadrado m e símbolo σ tais que:

1. Após n passos, M está no estado q examinando o quadrado m no qual aparece σ.

2. A função de transição $\delta(q,\sigma)$ é indefinida.

$C(M,w,n)$ é a descrição dessa configuração e incluirá as cláusulas $Q_q(\overline{m},\overline{n})$ e $S_\sigma(\overline{m},\overline{n})$. Essas cláusulas juntas implicam $E(M,w)$:

$$\exists x\, \exists y\, (\bigvee_{\langle q,\sigma\rangle \in X} (Q_q(x,y) \wedge S_\sigma(x,y)))$$

já que $Q_{q'}(\overline{m},\overline{n}) \wedge S_{\sigma'}(\overline{m},\overline{n}) \vDash \bigvee_{\langle q,\sigma\rangle \in X}(Q_q(\overline{m},\overline{n}) \wedge S_\sigma(\overline{m},\overline{n}))$, pois $\langle q',\sigma'\rangle \in X$. □

Assim, se M para na entrada w, então há algum n tal que $C(M,w,n) \vDash E(M,w)$. Mostraremos agora que para qualquer instante n, $T(M,w) \vDash C(M,w,n)$.

Lema 15.13. *Para cada n, se M não parou após n passos,* $T(M,w) \vDash C(M,w,n)$.

Prova. Base da indução: Se $n = 0$, então as fórmulas componentes de $C(M,w,0)$ são também fórmulas componentes de $T(M,w)$, logo implicadas por ela.

Hipótese de indução: Se M não parou antes do n-ésimo passo, então $T(M,w) \vDash C(M,w,n)$. Temos de mostrar que (a menos que $C(M,w,n)$ descreva uma configuração de parada), $T(M,w) \vDash C(M,w,n+1)$.

Suponha $n > 0$ e que, após n passos, M iniciada em w esteja no estado q examinando o quadrado m. Como M não para após n passos, deve haver no programa de M uma instrução de uma das três formas seguintes:

1. $\delta(q,\sigma) = \langle q',\sigma',R\rangle$
2. $\delta(q,\sigma) = \langle q',\sigma',L\rangle$
3. $\delta(q,\sigma) = \langle q',\sigma',N\rangle$

Consideraremos cada um desses três casos por sua vez.

1. Suponha que haja uma instrução da forma (1). Por Definição 15.9(3a), isso significa que

$$\forall x\, \forall y\, ((Q_q(x,y) \wedge S_\sigma(x,y)) \rightarrow (Q_{q'}(x',y') \wedge S_{\sigma'}(x,y') \wedge A(x,y)))$$

é uma fórmula componente de $T(M,w)$. Isso implica a seguinte sentença (instanciação universal, $\overline{m}$ por x e $\overline{n}$ por y):

$$(Q_q(\overline{m},\overline{n}) \wedge S_\sigma(\overline{m},\overline{n})) \rightarrow (Q_{q'}(\overline{m}',\overline{n}') \wedge S_{\sigma'}(\overline{m},\overline{n}') \wedge A(\overline{m},\overline{n})).$$

Por hipótese de indução, $T(M,w) \vDash C(M,w,n)$, isto é,

$$Q_q(\overline{m},\overline{n}) \wedge S_{\sigma_0}(\overline{0},\overline{n}) \wedge \cdots \wedge S_{\sigma_k}(\overline{k},\overline{n}) \wedge \forall x\, (\overline{k} < x \rightarrow S_\sqcup(x,\overline{n}))$$

Como após n passos o quadrado de fita m contém σ, a fórmula componente correspondente é $S_\sigma(\overline{m},\overline{n})$, de modo que isso implica:

$$Q_q(\overline{m},\overline{n}) \wedge S_\sigma(\overline{m},\overline{n})$$

Obtemos agora

$$Q_{q'}(\overline{m}',\overline{n}') \wedge S_{\sigma'}(\overline{m},\overline{n}') \wedge S_{\sigma_0}(\overline{0},\overline{n}') \wedge \cdots \wedge S_{\sigma_k}(\overline{k},\overline{n}') \wedge \forall x\, (\overline{k} < x \rightarrow S_\sqcup(x,\overline{n}'))$$

como segue: A primeira linha vem diretamente do consequente do condicional precedente, por modus ponens. Cada fórmula componente na linha do meio—que exclui $S_{\sigma_m}(\overline{m},\overline{n}')$—decorre da fórmula componente correspondente em $C(M,w,n)$ junto com $A(\overline{m},\overline{n})$.

Se $m < k$, $T(M,w) \vdash \overline{m} < \overline{k}$ (Proposição 15.10) e, por transitividade de $<$, temos $\forall x\,(\overline{k} < x \to \overline{m} < x)$. Se $m = k$, então $\forall x\,(\overline{k} < x \to \overline{m} < x)$ apenas por lógica. A última linha decorre então da fórmula componente correspondente em $C(M,w,n)$, de $\forall x\,(\overline{k} < x \to \overline{m} < x)$ e de $A(\overline{m},\overline{n})$. Se $m < k$, isto já é $C(M,w,n+1)$.

Agora suponha $m = k$. Nesse caso, após $n+1$ passos, o cabeçote de fita também visitou o quadrado $k+1$, que agora é o quadrado mais à direita visitado. Assim, $C(M,w,n+1)$ tem uma nova fórmula componente, $S_\sqcup(\overline{k}',\overline{n}')$, e a última fórmula componente é $\forall x\,(\overline{k}' < x \to S_\sqcup(x,\overline{n}'))$. Temos de verificar que essas duas sentenças também são implicadas.

Já temos $\forall x\,(\overline{k} < x \to S_\sqcup(x,\overline{n}'))$. Em particular, isso nos dá $\overline{k} < \overline{k}' \to S_\sqcup(\overline{k}',\overline{n}')$. Do axioma $\forall x\, x < x'$ obtemos $\overline{k} < \overline{k}'$. Por modus ponens, segue-se $S_\sqcup(\overline{k}',\overline{n}')$.

Além disso, como $T(M,w) \vdash \overline{k} < \overline{k}'$, o axioma da transitividade de $<$ nos dá $\forall x\,(\overline{k}' < x \to S_\sqcup(x,\overline{n}'))$. (Deixamos a verificação disso como exercício.)

2. Suponha que haja uma instrução da forma (2). Então, por Definição 15.9(3b),

$$\begin{aligned}
&\forall x\,\forall y\,((Q_q(x',y) \wedge S_\sigma(x',y)) \to \\
&\qquad (Q_{q'}(x,y') \wedge S_{\sigma'}(x',y') \wedge A(x,y))) \wedge \\
&\forall y\,((Q_{q_i}(\overline{0},y) \wedge S_\sigma(\overline{0},y)) \to \\
&\qquad (Q_{q_j}(\overline{0},y') \wedge S_{\sigma'}(\overline{0},y') \wedge A(\overline{0},y)))
\end{aligned}$$

é uma fórmula componente de $T(M,w)$. Se $m > 0$, então seja $l = m-1$ (isto é, $m = l+1$). A primeira fórmula componente da sentença acima implica o seguinte:

$$\begin{aligned}
&(Q_q(\overline{l}',\overline{n}) \wedge S_\sigma(\overline{l}',\overline{n})) \to \\
&\qquad (Q_{q'}(\overline{l},\overline{n}') \wedge S_{\sigma'}(\overline{l}',\overline{n}') \wedge A(\overline{l},\overline{n}))
\end{aligned}$$

Caso contrário, seja $l = m = 0$ e considere a seguinte sentença implicada pela segunda fórmula componente:

$$((Q_{q_i}(\overline{0},\overline{n}) \wedge S_\sigma(\overline{0},\overline{n})) \rightarrow (Q_{q_j}(\overline{0},\overline{n}') \wedge S_{\sigma'}(\overline{0},\overline{n}') \wedge A(\overline{0},\overline{n})))$$

Qualquer das sentenças implica

$$Q_{q'}(\overline{l},\overline{n}') \wedge S_{\sigma'}(\overline{m},\overline{n}') \wedge S_{\sigma_0}(\overline{0},\overline{n}') \wedge \cdots \wedge S_{\sigma_k}(\overline{k},\overline{n}') \wedge \forall x\,(\overline{k} < x \rightarrow S_\sqcup(x,\overline{n}'))$$

como antes. (Note que no primeiro caso, $\overline{l}' \equiv \overline{l+1} \equiv \overline{m}$ e no segundo caso $\overline{l} \equiv \overline{0}$.) Mas isto é justamente $C(M,w,n+1)$.

3. O caso (3) é deixado como exercício.

Mostramos que para qualquer n, $T(M,w) \vDash C(M,w,n)$. □

Lema 15.14. *Se M para na entrada w, então $T(M,w) \rightarrow E(M,w)$ é válida.*

Prova. Por Lema 15.13, sabemos que, para qualquer instante n, a descrição $C(M,w,n)$ da configuração de M no instante n é implicada por $T(M,w)$. Suponha que M pare após k passos. Nesse ponto, ela estará examinando o quadrado m, para algum $m \in \mathbb{N}$. Então $C(M,w,k)$ descreve uma configuração de parada de M, isto é, contém como fórmulas componentes tanto $Q_q(\overline{m},\overline{k})$ quanto $S_\sigma(\overline{m},\overline{k})$ com $\delta(q,\sigma)$ indefinida. Assim, por Lema 15.12, $C(M,w,k) \vDash E(M,w)$. Mas como $T(M,w) \vDash C(M,w,k)$, temos $T(M,w) \vDash E(M,w)$ e portanto $T(M,w) \rightarrow E(M,w)$ é válida. □

Para completar a verificação de nossa afirmação, temos também de estabelecer a direção reversa: se $T(M,w) \rightarrow E(M,w)$ é válida, então M de fato para quando iniciada na entrada w.

Lema 15.15. *Se* $\vDash T(M,w) \rightarrow E(M,w)$, *então* M *para na entrada* w.

Prova. Considere a $\mathscr{L}_M$-estrutura $\boldsymbol{M}$ com domínio $\mathbb{N}$ que interpreta o como 0, $\mathsf{'}$ como a função sucessor e $<$ como a relação menor-que, e os predicados Q_q e S_σ como segue:

$$Q_q^{\boldsymbol{M}} = \{\langle m,n\rangle : \begin{array}{l}\text{iniciada em } w, \text{ após } n \text{ passos,}\\ M \text{ está no estado } q \text{ examinando o quadrado } m\end{array}\}$$

$$S_\sigma^{\boldsymbol{M}} = \{\langle m,n\rangle : \begin{array}{l}\text{iniciada em } w, \text{ após } n \text{ passos,}\\ \text{o quadrado } m \text{ de } M \text{ contém o símbolo } \sigma\end{array}\}$$

Em outras palavras, construímos a estrutura $\boldsymbol{M}$ de modo que ela descreva o que M iniciada na entrada w de fato faz, passo a passo. Claramente, $\boldsymbol{M} \vDash T(M,w)$. Se $\vDash T(M,w) \rightarrow E(M,w)$, então também $\boldsymbol{M} \vDash E(M,w)$, isto é,

$$\boldsymbol{M} \vDash \exists x\, \exists y\, (\bigvee_{\langle q,\sigma\rangle \in X} (Q_q(x,y) \wedge S_\sigma(x,y))).$$

Como $|\boldsymbol{M}| = \mathbb{N}$, deve haver $m, n \in \mathbb{N}$ tais que $\boldsymbol{M} \vDash Q_q(\overline{m},\overline{n}) \wedge S_\sigma(\overline{m},\overline{n})$ para algum q e σ tais que $\delta(q,\sigma)$ é indefinida. Pela definição de $\boldsymbol{M}$, isso significa que M iniciada na entrada w após n passos está no estado q e lendo o símbolo σ, e a função de transição é indefinida, isto é, M parou. □

15.8 O Problema da Decisão é Insolúvel

Teorema 15.16. *O problema da decisão é insolúvel: Não há máquina de Turing* D *tal que, quando iniciada em uma fita que contém como entrada uma sentença* B *da lógica de primeira ordem,* D *eventualmente para e produz* 1 *se, e somente se,* B *é válida, e* 0 *caso contrário.*

Prova. Suponha que o problema da decisão fosse solúvel, isto é, suponha que houvesse uma máquina de Turing D. Então poderíamos resolver o problema da parada como segue. Construímos uma máquina de Turing E que, dado como entrada o número e

da máquina de Turing M_e e a entrada w, computa a sentença correspondente $T(M_e, w) \to E(M_e, w)$ e para, examinando o quadrado mais à esquerda na fita. A máquina $E \frown D$ então, dada a entrada e e w, primeiro computaria $T(M_e, w) \to E(M_e, w)$ e depois executaria a máquina do problema da decisão D nessa entrada. D para com saída 1 se, e somente se, $T(M_e, w) \to E(M_e, w)$ é válida, e produz 0 caso contrário. Por Lema 15.15 e Lema 15.14, $T(M_e, w) \to E(M_e, w)$ é válida se, e somente se, M_e para na entrada w. Assim, $E \frown D$, dada a entrada e e w, para com saída 1 se, e somente se, M_e para na entrada w, e para com saída 0 caso contrário. Em outras palavras, $E \frown D$ resolveria o problema da parada. Mas sabemos, por Teorema 15.8, que não pode existir tal máquina de Turing. □

Corolário 15.17. *É indecidível se uma sentença arbitrária da lógica de primeira ordem é satisfatível.*

Prova. Suponha que a satisfatibilidade fosse decidível por uma máquina de Turing S. Então poderíamos resolver o problema da decisão como segue: Dada como entrada uma sentença B, mova B um quadrado para a direita. Volte ao quadrado 1 e escreva o símbolo $\neg$.

Agora execute a máquina de Turing S. Ela eventualmente para com saída 1 (se $\neg B$ é satisfatível) ou 0 (se $\neg B$ é insatisfatível) na fita. Se há um I no quadrado 1, apague-o; se o quadrado 1 está vazio, escreva um I e então pare.

Essa máquina de Turing sempre para, e sua saída é 1 se, e somente se, $\neg B$ é insatisfatível, e 0 caso contrário. Como B é válida se, e somente se, $\neg B$ é insatisfatível, a máquina produz 1 se, e somente se, B é válida, e 0 caso contrário, isto é, ela resolveria o problema da decisão. □

Portanto, não há máquina de Turing que sempre dê uma resposta "sim" ou "não" correta à pergunta "B é uma sentença válida da lógica de primeira ordem?" No entanto, *há* uma máquina

de Turing que sempre dá uma resposta "sim" correta—mas que simplesmente não para se a resposta é "não". Isso decorre do teorema da correção e completude da lógica de primeira ordem, e do fato de que derivações podem ser efetivamente enumeradas.

Teorema 15.18. *A validade de sentenças de primeira ordem é semidecidível: Há uma máquina de Turing E que, quando iniciada em uma fita que contém como entrada uma sentença B da lógica de primeira ordem, E eventualmente para e produz* 1 *se, e somente se, B é válida, mas não para caso contrário.*

Prova. Todas as derivações possíveis da lógica de primeira ordem podem ser geradas, uma após a outra, por um algoritmo efetivo. A máquina E faz isso e, quando encontra uma derivação que mostra que $\vdash B$, ela para com saída 1. Pelo teorema da correção, se E para com saída 1, é porque $\vDash B$. Pelo teorema da completude, se $\vDash B$, há uma derivação que mostra que $\vdash B$. Como E gera sistematicamente todas as derivações possíveis, ela eventualmente encontrará uma que mostra $\vdash B$, de modo que eventualmente parará com saída 1. □

15.9 Teorema de Trakhtenbrot

Na seção 15.6 definimos as sentenças $T(M,w)$ e $E(M,w)$ para uma máquina de Turing M e uma cadeia de entrada w. Em seguida, mostramos em Lema 15.14 e Lema 15.15 que $T(M,w) \to E(M,w)$ é válida se, e somente se, M, iniciada na entrada w, eventualmente para. Como o Problema da Parada é indecidível, isso implica que a validade e a satisfatibilidade de sentenças da lógica de primeira ordem são indecidíveis (Teorema 15.16 e Corolário 15.17).

Mas a validade e a satisfatibilidade de sentenças são definidas para estruturas arbitrárias, finitas ou infinitas. Você poderia suspeitar que é mais fácil decidir se uma sentença é satisfatível em uma estrutura finita (ou válida em todas as estruturas fini-

tas). Podemos adaptar a prova da insolubilidade do problema da decisão de modo que ela mostre que não é esse o caso.

Primeiro, se você voltar à prova de Lema 15.15, verá que o que fizemos ali foi produzir um modelo $\mathcal{M}$ de $T(M,w)$ que descreve exatamente o que a máquina M faz quando iniciada na entrada w. O domínio daquele modelo era $\mathbb{N}$, isto é, infinito. Mas se M de fato para na entrada w, podemos construir um modelo finito $\mathcal{M}'$ da mesma maneira. Suponha que M iniciada na entrada w pare após k passos. Tome como domínio $|\mathcal{M}'|$ o conjunto $\{0,\ldots,n\}$, em que n é o maior entre k e o comprimento de w, e seja

$$\prime^{\mathcal{M}'}(x) = \begin{cases} x+1 & \text{se } x < n \\ n & \text{caso contrário,} \end{cases}$$

e $\langle x,y\rangle \in <^{\mathcal{M}'}$ se, e somente se, $x < y$ ou $x = y = n$. Quanto ao resto, $\mathcal{M}'$ é definida tal como $\mathcal{M}$. Pela definição de $\mathcal{M}'$, tal como na prova de Lema 15.15, $\mathcal{M}' \vDash T(M,w)$. E como supusemos que M para na entrada w, $\mathcal{M}' \vDash E(M,w)$. Assim, $\mathcal{M}'$ é um modelo finito de $T(M,w) \wedge E(M,w)$ (note que substituímos $\rightarrow$ por $\wedge$).

Estamos a meio caminho de uma prova: mostramos que se M para na entrada w, então $T(M,w) \wedge E(M,w)$ tem um modelo finito. Infelizmente, a recíproca disso não vale, isto é, há máquinas de Turing que não param em alguma entrada w, mas $T(M,w) \wedge E(M,w)$ ainda assim tem um modelo finito. Por exemplo, considere a máquina M com o único estado q_0 e a instrução $\delta(q_0,\sqcup) = \langle q_0,\sqcup,N\rangle$. Iniciada na entrada vazia $w = \Lambda$, essa máquina nunca para: está em um laço infinito, mas não altera a fita nem move o cabeçote. Todas as configurações são as mesmas (mesmo estado, mesma posição do cabeçote, mesmo conteúdo da fita). Podemos definir uma estrutura finita $\mathcal{M}''$ que satisfaz $T(M,\Lambda) \wedge E(M,\Lambda)$ (exercício). Podemos, no entanto, modificar $T(M,w)$ de modo adequado para que tais estruturas sejam descartadas.

Considere as sentenças que descrevem a operação da máquina de Turing M na entrada $w = \sigma_{i_1} \ldots \sigma_{i_k}$:

1. Axiomas que descrevem números e $<$ (tal como na definição de $T(M,w)$ na seção 15.6).

2. Axiomas que descrevem a configuração de entrada: tal como na definição de $T(M,w)$.

3. Axiomas que descrevem a transição de uma configuração para a seguinte:

   Para o que segue, seja $A(x,y)$ como antes, e seja

   $$B(y) \equiv \forall x\,(x < y \rightarrow x \neq y).$$

   a) Para toda instrução $\delta(q_i,\sigma) = \langle q_j,\sigma',R\rangle$, a sentença:

   $$\forall x\,\forall y\,((Q_{q_i}(x,y) \wedge S_\sigma(x,y)) \rightarrow (Q_{q_j}(x',y') \wedge S_{\sigma'}(x,y') \wedge A(x,y) \wedge B(y')))$$

   b) Para toda instrução $\delta(q_i,\sigma) = \langle q_j,\sigma',L\rangle$, a sentença

   $$\begin{aligned}&\forall x\,\forall y\,((Q_{q_i}(x',y) \wedge S_\sigma(x',y)) \rightarrow \\ &\qquad (Q_{q_j}(x,y') \wedge S_{\sigma'}(x',y') \wedge A(x,y))) \wedge \\ &\forall y\,((Q_{q_i}(\overline{0},y) \wedge S_\sigma(\overline{0},y)) \rightarrow \\ &\qquad (Q_{q_j}(\overline{0},y') \wedge S_{\sigma'}(\overline{0},y') \wedge A(\overline{0},y) \wedge B(y')))\end{aligned}$$

   c) Para toda instrução $\delta(q_i,\sigma) = \langle q_j,\sigma',N\rangle$, a sentença:

   $$\forall x\,\forall y\,((Q_{q_i}(x,y) \wedge S_\sigma(x,y)) \rightarrow (Q_{q_j}(x,y') \wedge S_{\sigma'}(x,y') \wedge A(x,y) \wedge B(y')))$$

   Como você pode ver, as sentenças que descrevem as transições de M são as mesmas que a sentença correspondente em $T(M,w)$, exceto que adicionamos $B(y')$ ao final. $B(y')$ garante que o número y' da "próxima" configuração é diferente de todos os números anteriores 0, $0'$, ...

Seja $T'(M,w)$ a conjunção de todas as sentenças acima para a máquina de Turing M e a entrada w.

Lema 15.19. *Se M iniciada na entrada w para, então $T'(M,w) \wedge E(M,w)$ tem um modelo finito.*

Prova. Seja M' como na prova de Lema 15.15, exceto que

$$|M'| = \{0,\ldots,n\},$$
$$s^{M'}(x) = \begin{cases} x+1 & \text{se } x < n \\ n & \text{caso contrário,} \end{cases}$$
$$\langle x,y\rangle \in <^{M'} \text{ sse } x < y \text{ ou } x = y = n,$$

em que $n = \max(k, \text{len}(w))$ e k é o menor número tal que M iniciada na entrada w parou após k passos. Deixamos como exercício a verificação de que $M' \vDash T'(M,w) \wedge E(M,w)$. □

Lema 15.20. *Se $T'(M,w) \wedge E(M,w)$ tem um modelo finito, então M iniciada na entrada w para.*

Prova. Mostramos a contrapositiva. Suponha que M iniciada em w não pare. Se $T'(M,w) \wedge E(M,w)$ não tem modelo algum, terminamos. Então suponha que M seja um modelo de $T(M,w) \wedge E(M,w)$. Temos de mostrar que ele não pode ser finito.

Podemos provar, tal como em Lema 15.13, que se M, iniciada na entrada w, não parou após n passos, então $T'(M,w) \vDash C(M,w,n) \wedge B(\overline{n})$. Como M iniciada na entrada w não para, $T'(M,w) \vDash C(M,w,n) \wedge B(\overline{n})$ para todo $n \in \mathbb{N}$. Note que, por Proposição 15.10, $T'(M,w) \vDash \overline{k} < \overline{n}$ para todo $k < n$. Além disso, $B(\overline{n}) \vDash \overline{k} < \overline{n} \rightarrow \overline{k} \neq \overline{n}$. Assim, $M \vDash \overline{k} \neq \overline{n}$ para todo $k < n$, isto é, os infinitos termos $\overline{k}$ devem ter todos valores diferentes em M. Mas isso requer que $|M|$ seja infinito, de modo que M não pode ser um modelo finito de $T'(M,w) \wedge E(M,w)$. □

Teorema 15.21 (Teorema de Trakhtenbrot). *É indecidível se uma sentença arbitrária da lógica de primeira ordem tem um modelo finito (isto é, é finitamente satisfatível).*

Prova. Suponha que houvesse uma máquina de Turing F que decide o problema da satisfatibilidade finita. Então, dada qualquer máquina de Turing M e entrada w, poderíamos computar a sentença $T'(M,w) \wedge E(M,w)$ e usar F para decidir se ela tem um modelo finito. Por Lemas 15.19 e 15.20, ela tem se, e somente se, M iniciada na entrada w para. Assim, poderíamos usar F para resolver o problema da parada, que sabemos ser insolúvel. □

Corolário 15.22. *Não pode haver um sistema de derivação que seja correto e completo para a validade* finita*, isto é, um sistema de derivação que tenha $\vdash B$ se, e somente se, $\mathbf{M} \models B$ para toda estrutura finita $\mathbf{M}$.*

Prova. Exercício. □

Resumo

Máquinas de Turing são determinadas por seus conjuntos de instruções, que são conjuntos finitos de quíntuplas (para cada estado e símbolo lido, especificam novo estado, símbolo escrito e movimento da cabeça). Os conjuntos finitos de quíntuplas são enumeráveis, de modo que há uma maneira de associar um número a cada conjunto de instruções de máquina de Turing. O **índice** de uma máquina de Turing é o número associado a seu conjunto de instruções sob um esquema fixo desse tipo. Dessa forma, podemos "falar sobre" máquinas de Turing indiretamente—falando sobre seus índices.

Um problema importante sobre o comportamento das máquinas de Turing é se elas eventualmente param. Seja $h(e,n)$ a função que = 1 se a máquina de Turing com índice e para quando iniciada com a entrada n, e = 0 caso contrário. Ela é chamada de **função de parada**. A questão de saber se a própria função

de parada é Turing-computável é chamada de **problema da parada**. A resposta é não: o problema da parada é insolúvel. Isso é estabelecido usando um argumento diagonal.

O problema da parada é apenas um exemplo de uma classe maior de problemas da forma "X pode ser realizado usando máquinas de Turing". Outro problema central da lógica é o **problema da decisão para a lógica de primeira ordem**: existe uma máquina de Turing capaz de decidir se uma dada sentença é válida ou não? Esse famoso problema também foi resolvido negativamente: o problema da decisão é insolúvel. Isso é estabelecido por um argumento de redução: podemos associar a cada máquina de Turing M e entrada w uma sentença de primeira ordem $T(M,w) \to E(M,w)$ que é válida se e somente se M para quando iniciada com a entrada w. Se o problema da decisão fosse solúvel, poderíamos então usá-lo para resolver o problema da parada.

Problemas

Problema 15.1. Você consegue pensar em uma maneira de descrever máquinas de Turing que não exija que os estados e os símbolos do alfabeto sejam listados explicitamente? Você pode definir sua própria noção de máquina "padrão", mas diga algo sobre por que toda máquina de Turing pode ser computada por uma máquina "padrão" no seu novo sentido.

Problema 15.2. O problema da Parada com Três (3-Parada) é o problema de dar um procedimento de decisão para determinar se uma máquina de Turing escolhida arbitrariamente para ou não para uma entrada de três I em uma fita que, fora isso, está em branco. Prove que o problema 3-Parada é insolúvel.

Problema 15.3. Mostre que se o problema da parada é solúvel para pares de máquina de Turing e entrada M_e e n em que $e \neq n$, então ele também é solúvel para os casos em que $e = n$.

Problema 15.4. Provamos que o problema da parada é insolúvel se a entrada é um número e, que identifica uma máquina de Turing M_e por meio de uma enumeração de todas as máquinas de Turing. E se permitirmos a descrição de máquinas de Turing de seção 15.2 diretamente como entrada? Pode haver uma máquina de Turing que decide o problema da parada, mas que recebe como entrada descrições de máquinas de Turing em vez de índices? Explique por que sim ou por que não.

Problema 15.5. Mostre que a função *parcial* s' definida como

$$s'(e) = \begin{cases} 1 & \text{se a máquina } M_e \text{ para na entrada } e \\ \text{indefinida} & \text{se a máquina } M_e \text{ não para na entrada } e \end{cases}$$

é Turing-computável.

Problema 15.6. Prove Proposição 15.10. (Dica: use indução em $k - m$).

Problema 15.7. Complete o caso (3) da prova de Lema 15.13.

Problema 15.8. Dê uma derivação de $S_{\sigma_i}(\overline{i}, \overline{n}')$ a partir de $S_{\sigma_i}(\overline{i}, \overline{n})$ e $A(m, n)$ (supondo $i \neq m$, isto é, $i < m$ ou $m < i$).

Problema 15.9. Dê uma derivação de $\forall x\,(\overline{k}' < x \rightarrow S_{\sqcup}(x, \overline{n}'))$ a partir de $\forall x\,(\overline{k} < x \rightarrow S_{\sqcup}(x, \overline{n}'))$, $\forall x\, x < x'$ e $\forall x \forall y \forall z\,((x < y \wedge y < z) \rightarrow x < z)$.)

Problema 15.10. Seja M uma máquina de Turing com o único estado q_0 e a única instrução $\delta(q_0, \sqcup) = \langle q, \sqcup, N\rangle$. Seja $|M''| = \{0, 1, 2\}$, $\nu^{M''}(0) = \nu^{M'}(1) = 1$ e $\nu^{M''}(2) = 2$, e $<^{M''} = \{\langle 0,1\rangle, \langle 1,1\rangle, \langle 2,2\rangle\}$. Defina $Q_{q_0}^{M''}$, $S_{\sqcup}^{M''}$ e $S_{\triangleright}^{M''}$ de modo que $T(M, \Lambda)$ e $E(M, \Lambda)$ se tornem verdadeiras e explique por que o são. Dica: Observe que $\delta(q_0, \triangleright)$ é indefinida. Garanta que

$$Q_{q_0}(\overline{1}, \overline{n}) \wedge S_{\triangleright}(\overline{0}, \overline{n}) \wedge \forall x\,(\overline{0} < x \rightarrow S_{\sqcup}(x, \overline{n})) \quad \text{para todo } n \in \mathbb{N}$$

$$\exists y\,(Q_{q_0}(\overline{0}, y) \wedge S_{\triangleright}(\overline{0}, y))$$

são ambas verdadeiras em M''.

Problema 15.11. Complete a prova de Lema 15.19 provando que $M' \vDash T(M,w) \wedge E(M,w)$.

Problema 15.12. Complete a prova de Lema 15.20 provando que se M, iniciada na entrada w, não parou após n passos, então $T'(M,w) \vDash B(\overline{n})$.

Problema 15.13. Prove Corolário 15.22. Observe que B é satisfeita em toda estrutura finita se, e somente se, $\neg B$ não é finitamente satisfatível. Explique por que a satisfatibilidade finita é semidecidível no sentido de Teorema 15.18. Use isso para argumentar que se houvesse um sistema de derivação para a validade finita, então a satisfatibilidade finita seria decidível.

Emmy Noether
1882 - 1935

APÊNDICE A

Provas

A.1 Introdução

Com base em suas experiências na lógica introdutória, você talvez se sinta à vontade com um sistema de derivação—provavelmente um sistema de derivação de dedução natural ou no estilo Fitch, ou talvez um sistema de árvores de prova. Você provavelmente se lembra de fazer provas nesses sistemas, seja provando uma fórmula, seja mostrando que um dado argumento é válido. Para fazer isso, você aplicava as regras do sistema até obter o resultado final desejado. Ao raciocinar *sobre* lógica, também provamos coisas, mas na maioria dos casos não estamos usando um sistema de derivação. De fato, a maior parte das provas que consideramos é feita em português (talvez com alguma linguagem simbólica inserida) e não inteiramente na linguagem da lógica de primeira ordem. Ao construir tais provas, você pode a princípio ficar perdido—como provo algo sem um sistema de derivação? Como começo? Como sei se minha prova está correta?

Antes de tentar uma prova, é importante saber o que é uma prova e como construir uma. Como o nome sugere, uma *prova* destina-se a mostrar que algo é verdadeiro. Você pode pensar nisso em termos de um diálogo—alguém lhe pergunta se algo é verdadeiro, digamos, se todo primo diferente de dois é um

número ímpar. Responder "sim" não é suficiente; a pessoa pode querer saber *por quê*. Nesse caso, você lhe daria uma prova.

No discurso cotidiano, pode ser suficiente esboçar uma resposta, ou dar uma resposta incompleta. Na lógica e na matemática, porém, queremos uma prova rigorosa—queremos mostrar que algo é verdadeiro além de *qualquer* dúvida. Isso significa que cada passo de nossa prova deve ser justificado, e a justificativa deve ser convincente (isto é, a suposição que você está usando é de fato assumida no enunciado do teorema que você está provando, as definições que você aplica devem ser aplicadas corretamente, as justificativas invocadas devem ser inferências corretas, etc.).

Em geral, estamos provando algum enunciado. Chamamos os enunciados que provamos por vários nomes: proposições, teoremas, lemas ou corolários. Uma proposição é um enunciado básico digno de prova: importante o suficiente para ser registrado, mas talvez não particularmente profundo nem aplicado com frequência. Um teorema é uma proposição significativa e importante. Sua prova é frequentemente dividida em vários passos, e às vezes recebe o nome da pessoa que primeiro o provou (por exemplo, o Teorema de Cantor, o teorema de Löwenheim–Skolem) ou do fato de que trata (por exemplo, o teorema da completude). Um lema é uma proposição ou teorema usado na prova de um resultado mais importante. De modo confuso, às vezes os lemas são resultados importantes em si mesmos, e também recebem o nome da pessoa que os introduziu (por exemplo, o Lema de Zorn). Um corolário é um resultado que decorre facilmente de outro.

Um enunciado a ser provado frequentemente contém suposições que esclarecem sobre quais tipos de coisas estamos provando algo. Ele pode começar com "Seja A uma fórmula da forma $B \to C$" ou "Suponha $\Gamma \vdash A$" ou algo do gênero. Essas são as *hipóteses* da proposição, teorema ou lema, e você pode assumi-las como verdadeiras em sua prova. Elas restringem o que estamos provando e também introduzem alguns nomes para os objetos sobre os quais estamos falando. Por exemplo, se sua

proposição começa com "Seja A uma fórmula da forma $B \to C$", você está provando algo apenas sobre todas as fórmulas de certo tipo (a saber, condicionais), e fica entendido que $B \to C$ é um condicional arbitrário sobre o qual sua prova falará.

A.2 Começando uma Prova

Mas por onde começar?

Foi-lhe dado algo a provar, de modo que isso deveria ser a última coisa mencionada na prova (você pode, obviamente, *anunciar* que vai prová-lo no começo, mas não convém usá-lo como suposição). Escreva o que você está tentando provar no rodapé de uma folha de papel em branco—assim você não perde de vista seu objetivo.

Em seguida, você pode ter algumas suposições que pode usar (isso ficará mais claro quando falarmos sobre o *tipo* de prova que você está fazendo na próxima seção). Escreva-as no topo da página e certifique-se de sinalizar que são suposições (isto é, se você está supondo p, escreva "suponha que p" ou "assuma que p"). Por fim, pode haver na questão algumas definições que você precisa conhecer. Pode ser que lhe digam para usar uma definição específica, ou pode haver várias definições nas suposições ou na conclusão para a qual você está trabalhando. *Anote-as e certifique-se de entender o que significam.*

A forma como você organiza sua prova também dependerá da forma da questão. A próxima seção fornece detalhes sobre como organizar sua prova com base no tipo de sentença.

A.3 Usando Definições

Mencionamos que você deve estar familiarizado com todas as definições que possam ser usadas na prova, e que saiba aplicá-las corretamente. Esse é um ponto realmente importante, e vale a pena examiná-lo em um pouco mais de detalhe. As definições são usadas para abreviar propriedades e relações, de modo que

possamos falar delas de forma mais sucinta. A abreviação introduzida é chamada de *definiendum*, e aquilo que ela abrevia é o *definiens*. Em provas, com frequência temos que voltar a como o definiendum foi introduzido, porque temos que explorar a estrutura lógica do definiens (a versão longa da qual o termo definido é a abreviação) para concluir nossa prova. Ao desdobrar as definições, você garante que está chegando ao cerne de onde a ação lógica acontece.

Começaremos com um exemplo. Suponha que você queira provar o seguinte:

Proposição A.1. *Para quaisquer conjuntos A e B, $A \cup B = B \cup A$.*

Para sequer iniciar a prova, precisamos saber o que significa que dois conjuntos sejam idênticos; isto é, precisamos saber o que o "=" naquela equação significa para conjuntos. Os conjuntos são definidos como idênticos sempre que têm os mesmos elementos. Assim, a definição que temos que desdobrar é:

Definição A.2. Os conjuntos A e B são *idênticos*, $A = B$, sse todo elemento de A é um elemento de B, e vice-versa.

Esta definição usa A e B como marcadores de posição para conjuntos arbitrários. O que ela define—o *definiendum*—é a expressão "$A = B$", dando a condição sob a qual $A = B$ é verdadeira. Essa condição—"todo elemento de A é um elemento de B, e vice-versa"—é o *definiens*.[1] A definição especifica que $A = B$ é verdadeira se, e somente se (abreviamos isso por "sse") a condição vale.

Quando você aplica a definição, tem que fazer corresponder o A e o B da definição ao caso de que está tratando. No nosso

[1] Neste caso particular—e de modo bastante confuso!—quando $A = B$, os conjuntos A e B são apenas um único e mesmo conjunto, ainda que usemos letras diferentes para ele do lado esquerdo e do lado direito. Mas os modos pelos quais esse conjunto é identificado podem ser diferentes, e isso torna a definição não trivial.

caso, isso significa que, para que $A \cup B = B \cup A$ seja verdadeira, cada $z \in A \cup B$ também deve estar em $B \cup A$, e vice-versa. A expressão $A \cup B$ na proposição faz o papel de A na definição, e $B \cup A$, o de B. Como A e B são usados tanto na definição quanto no enunciado da proposição que estamos provando, mas em usos diferentes, você precisa ter cuidado para não confundir os dois. Por exemplo, seria um erro pensar que você poderia provar a proposição mostrando que todo elemento de A é um elemento de B, e vice-versa—isso mostraria que $A = B$, não que $A \cup B = B \cup A$. (Além disso, como A e B podem ser quaisquer dois conjuntos, você não chegaria muito longe, pois, se nada for suposto sobre A e B, eles bem podem ser conjuntos diferentes.)

Dentro da prova, estamos lidando com noções da teoria dos conjuntos, como união, e portanto também precisamos conhecer os significados do símbolo $\cup$ a fim de entender como a prova deve prosseguir. E, às vezes, desdobrar a definição dá origem a outras definições a desdobrar. Por exemplo, $A \cup B$ é definido como $\{z : z \in A \text{ ou } z \in B\}$. Então, se você quer provar que $x \in A \cup B$, desdobrar a definição de $\cup$ lhe diz que você tem que provar $x \in \{z : z \in A \text{ ou } z \in B\}$. Agora você também tem que lembrar que $x \in \{z : \ldots z \ldots\}$ sse $\ldots x \ldots$. Assim, desdobrando ainda mais a definição da notação $\{z : \ldots z \ldots\}$, o que você tem que mostrar é: $x \in A$ ou $x \in B$. Assim, "todo elemento de $A \cup B$ é também um elemento de $B \cup A$" significa, na verdade: "para todo x, se $x \in A$ ou $x \in B$, então $x \in B$ ou $x \in A$." Se desdobramos completamente as definições na proposição, vemos que o que temos que mostrar é isto:

Proposição A.3. *Para quaisquer conjuntos A e B: (a) para todo x, se $x \in A$ ou $x \in B$, então $x \in B$ ou $x \in A$, e (b) para todo x, se $x \in B$ ou $x \in A$, então $x \in A$ ou $x \in B$.*

O que importa é que desdobrar definições é uma parte necessária da construção de uma prova. Fazê-lo corretamente é às vezes difícil: você deve ter cuidado para distinguir e fazer corresponder as variáveis da definição e os termos na afirmação

que está provando. Para ter sucesso, você deve saber o que a questão está pedindo e o que todos os termos usados na questão significam—muitas vezes você precisará desdobrar mais de uma definição. Em provas simples como as que seguem, a solução decorre quase imediatamente das próprias definições. É claro que nem sempre será assim tão simples.

A.4 Padrões de Inferência

As provas são compostas de inferências individuais. Quando fazemos uma inferência, normalmente a indicamos usando uma palavra como "então," "assim," ou "portanto." A inferência muitas vezes se apoia em um ou dois fatos que já temos disponíveis em nossa prova—pode ser algo que supusemos, ou algo que concluímos por uma inferência anterior. Para deixar claro, podemos rotular essas coisas, e na inferência indicamos quais outras afirmações estamos usando na inferência. Uma inferência muitas vezes também conterá uma explicação de *por que* nossa nova conclusão decorre das coisas que a precedem. Há alguns padrões comuns de inferência que são usados com muita frequência em provas; percorreremos alguns deles abaixo. Alguns padrões de inferência, como as provas por indução, são mais elaborados (e serão discutidos mais adiante).

Já discutimos um padrão de inferência: desdobrar, ou aplicar, uma definição. Quando desdobramos uma definição, apenas reenunciamos algo que envolve o definiendum usando o definiens. Por exemplo, suponha que já tenhamos estabelecido no curso de uma prova que $D = E$ (a). Então podemos aplicar a definição de = para conjuntos e inferir: "Assim, por definição, de (a), todo elemento de D é um elemento de E e vice-versa."

De modo um tanto confuso, muitas vezes não escrevemos a justificativa de uma inferência quando de fato a fazemos, mas antes. Suponha que ainda não tenhamos provado que $D = E$, mas queremos. Se $D = E$ é a conclusão que buscamos, então podemos reenunciar esse objetivo também aplicando a definição:

para provar $D = E$ temos que provar que todo elemento de D é um elemento de E e vice-versa. Assim, nossa prova terá a forma: (a) provar que todo elemento de D é um elemento de E; (b) todo elemento de E é um elemento de D; (c) portanto, de (a) e (b), pela definição de =, $D = E$. Mas normalmente não escreveríamos desse jeito. Em vez disso, poderíamos escrever algo como,

> Queremos mostrar $D = E$. Pela definição de =, isso equivale a mostrar que todo elemento de D é um elemento de E e vice-versa.
>
> (a) ... (uma prova de que todo elemento de D é um elemento de E) ...
>
> (b) ... (uma prova de que todo elemento de E é um elemento de D) ...

Usando uma Conjunção

Talvez o padrão de inferência mais simples seja o de tirar como conclusão uma das conjuntivas de uma conjunção. Em outras palavras: se supusemos ou já provamos que p e q, então temos o direito de inferir que p (e também que q). Essa é uma inferência tão básica que muitas vezes não é mencionada. Por exemplo, uma vez que tenhamos desdobrado a definição de $D = E$, estabelecemos que todo elemento de D é um elemento de E e vice-versa. Disso podemos concluir que todo elemento de E é um elemento de D (essa é a parte do "vice-versa").

Provando uma Conjunção

Às vezes, o que lhe pedirão para provar terá a forma de uma conjunção; pedirão a você que "prove p e q." Nesse caso, você simplesmente tem que fazer duas coisas: provar p, e então provar q. Você poderia dividir sua prova em duas seções e, para maior clareza, rotulá-las. Quando estiver fazendo suas primeiras anotações, você pode escrever "(1) Provar p" no topo da página, e "(2)

Provar q" no meio da página. (É claro que você pode não ser explicitamente solicitado a provar uma conjunção, mas descobrir que sua prova exige que você prove uma conjunção. Por exemplo, se lhe pedem para provar que $D = E$, você descobrirá que, após desdobrar a definição de =, você tem que provar: todo elemento de D é um elemento de E *e* todo elemento de E é um elemento de D).

Provando uma Disjunção

Quando o que você está provando assume a forma de uma disjunção (isto é, é uma afirmação da forma "p ou q"), basta mostrar que um dos disjuntos é verdadeiro. No entanto, basicamente nunca acontece de qualquer um dos disjuntos simplesmente decorrer das suposições do seu teorema. Mais frequentemente, as suposições do seu teorema são elas próprias disjuntivas, ou você está mostrando que todas as coisas de certo tipo têm uma de duas propriedades, mas algumas das coisas têm uma e outras têm a outra propriedade. É aqui que a prova por casos é útil (veja abaixo).

Prova Condicional

Muitos teoremas que você encontrará estão na forma condicional (isto é, mostre que se p vale, então q também é verdadeiro). Esses casos são bons e fáceis de montar—basta supor o antecedente do condicional (neste caso, p) e provar a conclusão q a partir dele. Então, se o seu teorema diz "Se p então q," você começa sua prova com "suponha p" e, ao final, você deve ter provado q.

Os condicionais podem ser enunciados de maneiras diferentes. Assim, em vez de "Se p então q," um teorema pode afirmar que "p somente se q," "q se p," ou "q, desde que p." Todos eles significam o mesmo e exigem supor p e provar q a partir dessa suposição. Lembre-se de que um bicondicional ("p se e somente se (sse) q") é, na verdade, dois condicionais postos juntos: se p então q, e se q então p. Tudo o que você tem que fazer, então,

são duas instâncias de prova condicional: uma para o primeiro condicional e outra para o segundo. Às vezes, no entanto, é possível provar uma afirmação "sse" encadeando um monte de outras afirmações "sse", de modo que você comece com "p" e termine com "q"—mas, nesse caso, você tem que garantir que cada passo realmente seja um "sse."

Afirmações Universais

Usar uma afirmação universal é simples: se algo é verdadeiro para qualquer coisa, é verdadeiro para cada coisa particular. Então, se, digamos, a hipótese da sua prova é $A \subseteq B$, isso significa (desdobrando a definição de $\subseteq$) que, para todo $x \in A$, $x \in B$. Assim, se você já sabe que $z \in A$, você pode concluir $z \in B$.

Provar uma afirmação universal pode parecer um pouco complicado. Normalmente essas afirmações assumem a seguinte forma: "Se x tem P, então ele tem Q" ou "Todos os P são Q." É claro que pode não se ajustar a essa forma perfeitamente, e é preciso um pouco de prática para descobrir o que exatamente lhe pedem para provar. Mas: muitas vezes temos que provar que todos os objetos com alguma propriedade têm uma certa outra propriedade.

A maneira de provar uma afirmação universal é introduzir nomes ou variáveis para as coisas que têm uma propriedade e então mostrar que elas também têm a outra propriedade. Poderíamos expressar isso dizendo que, para provar algo para *todos* os P, você tem que prová-lo para um P *arbitrário.* E o nome introduzido é um nome para um P arbitrário. Normalmente usamos letras isoladas como esses nomes para coisas arbitrárias, e as letras costumam seguir convenções: por exemplo, usamos n para números naturais, A para fórmulas, A para conjuntos, f para funções, etc.

O truque é manter a generalidade ao longo da prova. Você começa supondo que um objeto arbitrário ("x") tem a propriedade P, e mostra (com base apenas em definições ou no que lhe é permitido supor) que x tem a propriedade Q. Como você não

estipulou o que x é especificamente, a não ser que ele tem a propriedade P, então você pode afirmar que tudo que tem P tem a propriedade Q. Em suma, x é um substituto para *todas* as coisas com a propriedade P.

Proposição A.4. *Para todos os conjuntos A e B, $A \subseteq A \cup B$.*

Prova. Sejam A e B conjuntos arbitrários. Queremos mostrar que $A \subseteq A \cup B$. Pela definição de $\subseteq$, isso equivale a: para todo x, se $x \in A$ então $x \in A \cup B$. Então seja $x \in A$ um elemento arbitrário de A. Temos que mostrar que $x \in A \cup B$. Como $x \in A$, $x \in A$ ou $x \in B$. Assim, $x \in \{x : x \in A \vee x \in B\}$. Mas isso, pela definição de $\cup$, significa $x \in A \cup B$. □

Prova por Casos

Suponha que você tenha uma disjunção como suposição ou como conclusão já estabelecida—você supôs ou provou que p ou q é verdadeira. Você quer provar r. Você faz isso em dois passos: primeiro você supõe que p é verdadeira, e prova r, e depois supõe que q é verdadeira e prova r novamente. Isso funciona porque supomos ou sabemos que uma das duas alternativas vale. Os dois passos estabelecem que qualquer uma delas é suficiente para a verdade de r. (Se ambas são verdadeiras, temos não uma, mas duas razões para que r seja verdadeira. Não é necessário provar separadamente que r é verdadeira supondo tanto p quanto q.) Para indicar o que estamos fazendo, anunciamos que "distinguimos casos." Por exemplo, suponha que saibamos que $x \in B \cup C$. $B \cup C$ é definido como $\{x : x \in B \text{ ou } x \in C\}$. Em outras palavras, por definição, $x \in B$ ou $x \in C$. Provaríamos que $x \in A$ a partir disso supondo primeiro que $x \in B$, e provando $x \in A$ a partir dessa suposição, e então supondo $x \in C$, e novamente provando $x \in A$ a partir disso. Você escreveria "Distinguimos casos" sob a suposição, depois "Caso (1): $x \in B$" abaixo, e "Caso (2): $x \in C$" na metade da página. Então você prosseguiria preenchendo a metade superior e a metade inferior da página.

A prova por casos é especialmente útil se o que você está provando é, ele mesmo, disjuntivo. Eis um exemplo simples:

Proposição A.5. *Suponha $B \subseteq D$ e $C \subseteq E$. Então $B \cup C \subseteq D \cup E$.*

Prova. Suponha (a) que $B \subseteq D$ e (b) $C \subseteq E$. Por definição, qualquer $x \in B$ também está $\in D$ (c) e qualquer $x \in C$ também está $\in E$ (d). Para mostrar que $B \cup C \subseteq D \cup E$, temos que mostrar que se $x \in B \cup C$ então $x \in D \cup E$ (pela definição de $\subseteq$). $x \in B \cup C$ sse $x \in B$ ou $x \in C$ (pela definição de $\cup$). De modo semelhante, $x \in D \cup E$ sse $x \in D$ ou $x \in E$. Então, temos que mostrar: para qualquer x, se $x \in B$ ou $x \in C$, então $x \in D$ ou $x \in E$.

> Até aqui só desdobramos definições! Reformulamos nossa proposição sem $\subseteq$ e $\cup$ e ficamos tentando provar uma afirmação condicional universal. Pelo que discutimos acima, isso é feito supondo que x é algo a respeito do qual supomos que a parte do "se" é verdadeira, e passaremos a mostrar que a parte do "então" também é verdadeira. Em outras palavras, suporemos que $x \in B$ ou $x \in C$ e mostraremos que $x \in D$ ou $x \in E$.[2]

Suponha que $x \in B$ ou $x \in C$. Temos que mostrar que $x \in D$ ou $x \in E$. Distinguimos casos.

Caso 1: $x \in B$. Por (c), $x \in D$. Assim, $x \in D$ ou $x \in E$. (Aqui fizemos a inferência discutida na subseção anterior!)

Caso 2: $x \in C$. Por (d), $x \in E$. Assim, $x \in D$ ou $x \in E$. □

Provando uma Afirmação de Existência

Quando lhe pedem para provar uma afirmação de existência, a questão normalmente será da forma "prove que há um x tal que

[2] Este parágrafo apenas explica o que estamos fazendo—não é parte da prova, e você não precisa entrar em todo esse detalhe quando escreve suas próprias provas.

$\ldots x \ldots$", isto é, que algum objeto tem a propriedade descrita por "$\ldots x \ldots$". Nesse caso, você terá que identificar um objeto adequado e mostrar que ele tem a propriedade exigida. Isso parece direto, mas uma prova desse tipo pode ser complicada. Tipicamente, ela envolve *construir* ou *definir* um objeto e provar que o objeto assim definido tem a propriedade exigida. Encontrar o objeto certo pode ser difícil, provar que ele tem a propriedade exigida pode ser difícil, e às vezes é até complicado mostrar que você teve sucesso em definir um objeto!

Em geral, você escreveria isso especificando o objeto, por exemplo, "seja $x \ldots$" (em que $\ldots$ especifica qual objeto você tem em mente), possivelmente provando que $\ldots$ de fato descreve um objeto que existe, e então passaria a mostrar que x tem a propriedade Q. Eis um exemplo simples.

Proposição A.6. *Suponha que $x \in B$. Então há um A tal que $A \subseteq B$ e $A \neq \emptyset$.*

Prova. Suponha $x \in B$. Seja $A = \{x\}$.

> Aqui definimos o conjunto A enumerando seus elementos. Como supomos que x é um objeto, e sempre podemos formar um conjunto enumerando seus elementos, não temos que mostrar que tivemos sucesso em definir um conjunto A aqui. No entanto, ainda temos que mostrar que A tem as propriedades exigidas pela proposição. A prova não está completa sem isso!

Como $x \in A$, $A \neq \emptyset$.

> Isso se apoia na definição de A como $\{x\}$ e nos fatos óbvios de que $x \in \{x\}$ e $x \notin \emptyset$.

Como x é o único elemento de $\{x\}$, e $x \in B$, todo elemento de A é também um elemento de B. Pela definição de $\subseteq$, $A \subseteq B$. □

Usando Afirmações de Existência

Suponha que você saiba que alguma afirmação de existência é verdadeira (você a provou, ou ela é uma hipótese que você pode usar), digamos, "para algum x, $x \in A$" ou "há um $x \in A$." Se você quiser usá-la em sua prova, você pode simplesmente fingir que tem um nome para uma das coisas que sua hipótese diz existir. Como A contém pelo menos uma coisa, há coisas a que esse nome poderia se referir. É claro que você pode não ser capaz de escolher uma ou de descrevê-la mais (a não ser que ela é $\in A$). Mas, para os fins da prova, você pode fingir que a escolheu e dar-lhe um nome. É importante escolher um nome que você ainda não tenha usado (ou que apareça nas suas hipóteses), caso contrário as coisas podem dar errado. Em sua prova, você indica isso passando de "para algum x, $x \in A$" para "Seja $a \in A$." Agora você pode raciocinar sobre a, usar algumas outras hipóteses, etc., até chegar a uma conclusão, p. Se p não menciona mais a, p é independente da suposição de que $a \in A$, e você mostrou que ele decorre apenas da suposição "para algum x, $x \in A$."

Proposição A.7. *Se $A \neq \emptyset$, então $A \cup B \neq \emptyset$.*

Prova. Suponha $A \neq \emptyset$. Então, para algum x, $x \in A$.

> Aqui apenas reenunciamos a hipótese da proposição. Essa hipótese, isto é, $A \neq \emptyset$, esconde uma afirmação existencial, à qual você só chega desdobrando algumas definições. A definição de $=$ nos diz que $A = \emptyset$ sse todo $x \in A$ também está $\in \emptyset$ e todo $x \in \emptyset$ também está $\in A$. Negando ambos os lados, obtemos: $A \neq \emptyset$ sse ou algum $x \in A$ está $\notin \emptyset$ ou algum $x \in \emptyset$ está $\notin A$. Como nada está $\in \emptyset$, o segundo disjunto nunca pode ser verdadeiro, e "$x \in A$ e $x \notin \emptyset$" se reduz a apenas $x \in A$. Então $A \neq \emptyset$ sse, para algum x, $x \in A$. Essa é uma afirmação de existência. Agora usamos essa afirmação de existência introduzindo um nome para um dos elementos de A:

Seja $a \in A$.

> Agora introduzimos um nome para uma das coisas $\in A$. Continuaremos a argumentar sobre a, mas teremos o cuidado de supor apenas que $a \in A$ e nada mais:

Como $a \in A$, $a \in A \cup B$, pela definição de $\cup$. Então, para algum x, $x \in A \cup B$, isto é, $A \cup B \neq \emptyset$.

> Naquele último passo, fomos de "$a \in A \cup B$" para "para algum x, $x \in A \cup B$." Isso não menciona mais a, então sabemos que "para algum x, $x \in A \cup B$" decorre de "para algum x, $x \in A$" apenas. Mas isso significa que $A \cup B \neq \emptyset$. □

Talvez seja boa prática manter variáveis ligadas como "x" separadas de nomes hipotéticos como a, como fizemos. Na prática, porém, muitas vezes não o fazemos e apenas usamos x, assim:

> Suponha $A \neq \emptyset$, isto é, há um $x \in A$. Pela definição de $\cup$, $x \in A \cup B$. Então $A \cup B \neq \emptyset$.

No entanto, quando você faz isso, tem que ter um cuidado extra para usar x e y diferentes para afirmações existenciais diferentes. Por exemplo, a seguinte *não* é uma prova correta de "Se $A \neq \emptyset$ e $B \neq \emptyset$ então $A \cap B \neq \emptyset$" (o que não é verdade).

> Suponha $A \neq \emptyset$ e $B \neq \emptyset$. Então, para algum x, $x \in A$ e também, para algum x, $x \in B$. Como $x \in A$ e $x \in B$, $x \in A \cap B$, pela definição de $\cap$. Então $A \cap B \neq \emptyset$.

Você consegue identificar onde ocorre o passo incorreto e explicar por que o resultado não vale?

A.5 Um Exemplo

Nosso primeiro exemplo é o seguinte fato simples sobre uniões e interseções de conjuntos. Ele ilustrará o desdobramento de definições, provas de conjunções, de afirmações universais, e a prova por casos.

Proposição A.8. *Para quaisquer conjuntos A, B e C, $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$*

Vamos prová-la!

Prova. Queremos mostrar que, para quaisquer conjuntos A, B e C, $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

> Primeiro desdobramos a definição de "=" no enunciado da proposição. Lembre-se de que provar que conjuntos são idênticos significa mostrar que os conjuntos têm os mesmos elementos. Isto é, todos os elementos de $A \cup (B \cap C)$ são também elementos de $(A \cup B) \cap (A \cup C)$, e vice-versa. O "vice-versa" significa que também todo elemento de $(A \cup B) \cap (A \cup C)$ deve ser um elemento de $A \cup (B \cap C)$. Assim, ao desdobrar a definição, vemos que temos que provar uma conjunção. Vamos registrar isso:

Por definição, $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ sse todo elemento de $A \cup (B \cap C)$ é também um elemento de $(A \cup B) \cap (A \cup C)$, e todo elemento de $(A \cup B) \cap (A \cup C)$ é um elemento de $A \cup (B \cap C)$.

> Como isso é uma conjunção, devemos provar cada conjuntiva separadamente. Vamos começar pela primeira: vamos provar que todo elemento de $A \cup (B \cap C)$ é também um elemento de $(A \cup B) \cap (A \cup C)$.
>
> Esta é uma afirmação universal, e por isso consideramos um elemento arbitrário de $A \cup (B \cap C)$ e mostramos que ele também deve ser um elemento de

> $(A \cup B) \cap (A \cup C)$. Vamos escolher uma variável para chamar esse elemento arbitrário, digamos, z. Nossa prova continua:

Primeiro, provamos que todo elemento de $A \cup (B \cap C)$ é também um elemento de $(A \cup B) \cap (A \cup C)$. Seja $z \in A \cup (B \cap C)$. Temos que mostrar que $z \in (A \cup B) \cap (A \cup C)$.

> Agora é hora de desdobrar a definição de $\cup$ e $\cap$. Por exemplo, a definição de $\cup$ é: $A \cup B = \{z : z \in A \text{ ou } z \in B\}$. Quando aplicamos a definição a "$A \cup (B \cap C)$," o papel do "B" na definição passa a ser desempenhado por "$B \cap C$," de modo que $A \cup (B \cap C) = \{z : z \in A \text{ ou } z \in B \cap C\}$. Então nossa suposição de que $z \in A \cup (B \cap C)$ equivale a: $z \in \{z : z \in A \text{ ou } z \in B \cap C\}$. E $z \in \{z : \ldots z \ldots\}$ sse $\ldots z \ldots$, isto é, neste caso, $z \in A$ ou $z \in B \cap C$.

Pela definição de $\cup$, ou $z \in A$ ou $z \in B \cap C$.

> Como isso é uma disjunção, será útil aplicar a prova por casos. Tomamos os dois casos, e mostramos que, em cada um, a conclusão que buscamos (a saber, "$z \in (A \cup B) \cap (A \cup C)$") se obtém.

Caso 1: Suponha que $z \in A$.

> Não há muito mais com que trabalhar a partir das nossas suposições. Então vamos olhar para o que temos com que trabalhar na conclusão. Queremos mostrar que $z \in (A \cup B) \cap (A \cup C)$. Com base na definição de $\cap$, se queremos mostrar que $z \in (A \cup B) \cap (A \cup C)$, temos que mostrar que ele está tanto em $(A \cup B)$ quanto em $(A \cup C)$. Mas $z \in A \cup B$ sse $z \in A$ ou $z \in B$, e já temos (como a suposição do caso 1) que $z \in A$. Pelo mesmo raciocínio—trocando C por B—$z \in A \cup C$. Esse argumento foi na direção inversa, então vamos registrar nosso raciocínio na direção necessária à nossa prova.

Como $z \in A$, $z \in A$ ou $z \in B$, e logo, pela definição de $\cup$, $z \in A \cup B$. De modo semelhante, $z \in A \cup C$. Mas isso significa que $z \in (A \cup B) \cap (A \cup C)$, pela definição de $\cap$.

> Isso completa o primeiro caso da prova por casos. Agora queremos derivar a conclusão no segundo caso, em que $z \in B \cap C$.

Caso 2: Suponha que $z \in B \cap C$.

> Novamente, estamos trabalhando com a interseção de dois conjuntos. Vamos aplicar a definição de $\cap$:

Como $z \in B \cap C$, z deve ser um elemento tanto de B quanto de C, pela definição de $\cap$.

> É hora de olhar de novo para a nossa conclusão. Temos que mostrar que z está tanto em $(A \cup B)$ quanto em $(A \cup C)$. E, novamente, a solução é imediata.

Como $z \in B$, $z \in (A \cup B)$. Como $z \in C$, também $z \in (A \cup C)$. Então, $z \in (A \cup B) \cap (A \cup C)$.

> Aqui aplicamos as definições de $\cup$ e $\cap$ novamente, mas como já relembramos essas definições, e já mostramos que, se z está em um de dois conjuntos, ele está na união deles, não precisamos ser tão explícitos no que fizemos.
>
> Completamos o segundo caso da prova por casos, então agora podemos afirmar nossa primeira conclusão.

Então, se $z \in A \cup (B \cap C)$, então $z \in (A \cup B) \cap (A \cup C)$.

> Agora queremos apenas mostrar a outra direção, que todo elemento de $(A \cup B) \cap (A \cup C)$ é um elemento de $A \cup (B \cap C)$. Como antes, provamos essa afirmação universal supondo que temos um elemento arbitrário do primeiro conjunto e mostrando que ele deve estar

no segundo conjunto. Vamos declarar o que estamos prestes a fazer.

Agora, suponha que $z \in (A \cup B) \cap (A \cup C)$. Queremos mostrar que $z \in A \cup (B \cap C)$.

Estamos agora trabalhando a partir da hipótese de que $z \in (A \cup B) \cap (A \cup C)$. Esperamos que não seja confuso demais estarmos usando o mesmo z aqui que na primeira parte da prova. Quando terminamos aquela parte, todas as suposições que fizemos ali deixaram de estar em vigor, então agora podemos fazer novas suposições sobre o que z é. Se isso for confuso para você, basta substituir z por uma variável diferente no que segue.

Sabemos que z está tanto em $A \cup B$ quanto em $A \cup C$, pela definição de $\cap$. E, pela definição de $\cup$, podemos desdobrar isso ainda mais em: ou $z \in A$ ou $z \in B$, e também ou $z \in A$ ou $z \in C$. Isso parece uma prova por casos de novo—exceto que o "e" a torna confusa. Você poderia pensar que isso equivale a haver três possibilidades: z está ou em A, B ou C. Mas isso seria um erro. Temos que ter cuidado, então vamos considerar cada disjunção por vez.

Pela definição de $\cap$, $z \in A \cup B$ e $z \in A \cup C$. Pela definição de $\cup$, $z \in A$ ou $z \in B$. Distinguimos casos.

Como estamos nos concentrando na primeira disjunção, ainda não obtivemos nossa segunda disjunção (do desdobramento de $A \cup C$). De fato, ainda não precisamos dela. O primeiro caso é $z \in A$, e um elemento de um conjunto é também um elemento da união desse conjunto com qualquer outro. Então o caso 1 é fácil:

Caso 1: Suponha que $z \in A$. Segue-se que $z \in A \cup (B \cap C)$.

Agora, para o segundo caso, $z \in B$. Aqui desdobraremos o segundo $\cup$ e faremos outra prova por casos:

Caso 2: Suponha que $z \in B$. Como $z \in A \cup C$, ou $z \in A$ ou $z \in C$. Distinguimos os casos ainda mais:

Caso 2a: $z \in A$. Então, novamente, $z \in A \cup (B \cap C)$.

Ok, isso foi um pouco estranho. Na verdade, não precisamos da suposição de que $z \in B$ para este caso, mas tudo bem.

Caso 2b: $z \in C$. Então $z \in B$ e $z \in C$, de modo que $z \in B \cap C$, e consequentemente, $z \in A \cup (B \cap C)$.

Isso conclui ambas as provas por casos, e assim terminamos a segunda metade.

Então, se $z \in (A \cup B) \cap (A \cup C)$, então $z \in A \cup (B \cap C)$. □

A.6 Outro Exemplo

Proposição A.9. *Se $A \subseteq C$, então $A \cup (C \setminus A) = C$.*

Prova. Suponha que $A \subseteq C$. Queremos mostrar que $A \cup (C \setminus A) = C$.

Começamos observando que esta é uma afirmação condicional. Ela está tacitamente quantificada universalmente: a proposição vale para todos os conjuntos A e C. Assim, A e C são variáveis para conjuntos arbitrários. Para provar tal afirmação, supomos o antecedente e provamos o consequente.

Continuamos usando a suposição de que $A \subseteq C$. Vamos desdobrar a definição de $\subseteq$: a suposição significa que todos os elementos de A são também elementos de C. Vamos anotar isto—é um fato importante que usaremos ao longo da prova.

Pela definição de $\subseteq$, como $A \subseteq C$, para todo z, se $z \in A$, então $z \in C$.

> Desdobramos todas as definições que nos são dadas na suposição. Agora podemos passar à conclusão. Queremos mostrar que $A \cup (C \setminus A) = C$, e por isso montamos uma prova de modo semelhante ao último exemplo: mostramos que todo elemento de $A \cup (C \setminus A)$ é também um elemento de C e, reciprocamente, todo elemento de C é um elemento de $A \cup (C \setminus A)$. Podemos abreviar isso para: $A \cup (C \setminus A) \subseteq C$ e $C \subseteq A \cup (C \setminus A)$. (Aqui estamos fazendo o oposto de desdobrar uma definição, mas isso torna a prova um pouco mais fácil de ler.) Como isso é uma conjunção, temos que provar ambas as partes. Para mostrar a primeira parte, isto é, que todo elemento de $A \cup (C \setminus A)$ é também um elemento de C, supomos que $z \in A \cup (C \setminus A)$ para um z arbitrário e mostramos que $z \in C$. Pela definição de $\cup$, podemos concluir que $z \in A$ ou $z \in C \setminus A$ a partir de $z \in A \cup (C \setminus A)$. Você já deve estar pegando o jeito disso.

$A \cup (C \setminus A) = C$ sse $A \cup (C \setminus A) \subseteq C$ e $C \subseteq (A \cup (C \setminus A)$. Primeiro provamos que $A \cup (C \setminus A) \subseteq C$. Seja $z \in A \cup (C \setminus A)$. Então, ou $z \in A$ ou $z \in (C \setminus A)$.

> Chegamos a uma disjunção, e a partir dela queremos provar que $z \in C$. Fazemos isso usando a prova por casos.

Caso 1: $z \in A$. Como, para todo z, se $z \in A$, $z \in C$, temos que $z \in C$.

> Aqui usamos o fato registrado anteriormente, que decorreu da hipótese da proposição de que $A \subseteq C$. O primeiro caso está completo, e passamos ao segundo caso, $z \in (C \setminus A)$. Lembre-se de que $C \setminus A$ denota

a *diferença* dos dois conjuntos, isto é, o conjunto de todos os elementos de C que não são elementos de A. Mas qualquer elemento de C que não esteja em A é, em particular, um elemento de C.

Caso 2: $z \in (C \setminus A)$. Isso significa que $z \in C$ e $z \notin A$. Então, em particular, $z \in C$.

Ótimo, provamos a primeira direção. Agora a segunda direção. Aqui provamos que $C \subseteq A \cup (C \setminus A)$. Então supomos que $z \in C$ e provamos que $z \in A \cup (C \setminus A)$.

Agora seja $z \in C$. Queremos mostrar que $z \in A$ ou $z \in C \setminus A$.

Como todos os elementos de A são também elementos de C, e $C \setminus A$ é o conjunto de todas as coisas que são elementos de C mas não de A, segue-se que z está ou em A ou em $C \setminus A$. Isso pode ser um pouco obscuro se você ainda não souber por que o resultado é verdadeiro. Seria melhor prová-lo passo a passo. Ajudará usar um fato simples que podemos enunciar sem prova: $z \in A$ ou $z \notin A$. Isso é chamado de "princípio do terceiro excluído:" para qualquer afirmação p, ou p é verdadeira ou sua negação é verdadeira. (Aqui, p é a afirmação de que $z \in A$.) Como isso é uma disjunção, podemos novamente usar a prova por casos.

Ou $z \in A$ ou $z \notin A$. No primeiro caso, $z \in A \cup (C \setminus A)$. No segundo caso, $z \in C$ e $z \notin A$, de modo que $z \in C \setminus A$. Mas então $z \in A \cup (C \setminus A)$.

Nossa prova está completa: mostramos que $A \cup (C \setminus A) = C$. □

A.7 Prova por Contradição

Em primeira instância, a prova por contradição é um padrão de inferência usado para provar afirmações negativas. Suponha que você queira mostrar que alguma afirmação p é *falsa*, isto é, você quer mostrar $\neg p$. A estratégia mais promissora é (a) supor que p é verdadeira, e (b) mostrar que essa suposição leva a algo que você sabe ser falso. "Algo que se sabe ser falso" pode ser um resultado que entra em conflito com—contradiz—o próprio p, ou alguma outra hipótese da afirmação geral que você está considerando. Por exemplo, uma prova de "se q então $\neg p$" envolve supor que q é verdadeira e provar $\neg p$ a partir disso. Se você prova $\neg p$ por contradição, isso significa supor p além de q. Se você consegue provar $\neg q$ a partir de p, mostrou que a suposição p leva a algo que contradiz sua outra suposição q, já que q e $\neg q$ não podem ser ambas verdadeiras. É claro que você tem que usar outros padrões de inferência em sua prova da contradição, bem como desdobrar definições. Vamos considerar um exemplo.

Proposição A.10. *Se $A \subseteq B$ e $B = \emptyset$, então A não tem elementos.*

Prova. Suponha $A \subseteq B$ e $B = \emptyset$. Queremos mostrar que A não tem elementos.

> Como esta é uma afirmação condicional, supomos o antecedente e queremos provar o consequente. O consequente é: A não tem elementos. Podemos tornar isso um pouco mais explícito: não é o caso de que haja um $x \in A$.

A não tem elementos sse não é o caso de que haja um x tal que $x \in A$.

> Então determinamos que o que queremos provar é, na verdade, uma afirmação negativa $\neg p$, a saber: não é o caso de que haja um $x \in A$. Para usar a prova por contradição, temos que supor a afirmação positiva

> correspondente p, isto é, há um $x \in A$, e provar uma contradição a partir dela. Indicamos que estamos fazendo uma prova por contradição escrevendo "por absurdo, suponha" ou mesmo apenas "suponha que não," e então enunciamos a suposição p.

Suponha que não: há um $x \in A$.

> Esta é agora a nova suposição que usaremos para obter uma contradição. Temos mais duas suposições: que $A \subseteq B$ e que $B = \emptyset$. A primeira nos dá que $x \in B$:

Como $A \subseteq B$, $x \in B$.

> Mas como $B = \emptyset$, todo elemento de B (por exemplo, x) também deve ser um elemento de $\emptyset$.

Como $B = \emptyset$, $x \in \emptyset$. Isso é uma contradição, já que, por definição, $\emptyset$ não tem elementos.

> Isso já completa a prova: chegamos ao que precisávamos (uma contradição) a partir das suposições que montamos, e isso significa que as suposições não podem ser todas verdadeiras. Como as duas primeiras suposições ($A \subseteq B$ e $B = \emptyset$) não são contestadas, deve ser a última suposição introduzida (há um $x \in A$) que tem que ser falsa. Mas, se quisermos ser minuciosos, podemos explicitar isso.

Assim, nossa suposição de que há um $x \in A$ tem que ser falsa, logo, A não tem elementos, por prova por contradição. □

Toda afirmação positiva é trivialmente equivalente a uma afirmação negativa: p sse $\neg\neg p$. Assim, as provas por contradição também podem ser usadas para estabelecer afirmações positivas "indiretamente," como segue: Para provar p, leia-o como a afirmação negativa $\neg\neg p$. Se conseguirmos provar uma contradição a partir de $\neg p$, estabelecemos $\neg\neg p$ por prova por contradição, e portanto p.

No último exemplo, buscávamos provar uma afirmação negativa, a saber, que A não tem elementos, e por isso a suposição que fizemos para os fins da prova por contradição (isto é, que há um $x \in A$) era uma afirmação positiva. Ela nos deu algo com que trabalhar, a saber, o hipotético $x \in A$ sobre o qual continuamos a raciocinar até chegarmos a $x \in \emptyset$.

Ao provar uma afirmação positiva indiretamente, a suposição que você faria para os fins da prova por contradição seria negativa. Mas muito frequentemente você pode facilmente reformular uma afirmação positiva como uma afirmação negativa, e uma afirmação negativa como uma afirmação positiva. Nossa prova anterior teria sido essencialmente a mesma se tivéssemos provado "$A = \emptyset$" em vez do consequente negativo "A não tem elementos." (Pela definição de $=$, "$A = \emptyset$" é uma afirmação geral, já que ela se desdobra em "todo elemento de A é um elemento de $\emptyset$ e vice-versa".) Mas é facilmente visto que ela é equivalente à afirmação negativa "não: há um $x \in A$."

Então, às vezes é mais fácil trabalhar com $\neg p$ como suposição do que provar p diretamente. Mesmo quando uma prova direta é igualmente simples ou até mais simples (como nos próximos exemplos), algumas pessoas preferem proceder indiretamente. Se a dupla negação o confunde, pense em uma prova por contradição de alguma afirmação como uma prova de uma contradição a partir da afirmação *oposta*. Assim, uma prova por contradição de $\neg p$ é uma prova de uma contradição a partir da suposição p; e a prova por contradição de p é uma prova de uma contradição a partir de $\neg p$.

Proposição A.11. $A \subseteq A \cup B$.

Prova. Queremos mostrar que $A \subseteq A \cup B$.

> À primeira vista, esta é uma afirmação positiva: todo $x \in A$ está também em $A \cup B$. A negação disso é: algum $x \in A$ está $\notin A \cup B$. Então podemos provar a afirmação indiretamente supondo essa afirmação negada, e mostrando que ela leva a uma contradição.

Suponha que não, isto é, $A \nsubseteq A \cup B$.

> Temos uma definição de $A \subseteq A \cup B$: todo $x \in A$ está também $\in A \cup B$. Para entender o que $A \nsubseteq A \cup B$ significa, temos que usar alguma manipulação lógica elementar sobre a definição desdobrada: é falso que todo $x \in A$ esteja também $\in A \cup B$ sse há *algum* $x \in A$ que está $\notin C$. (Este é um ponto em que você quer ser muito cuidadoso: muitas tentativas de prova por contradição de estudantes falham porque eles analisam a negação de uma afirmação como "todos os A são B" incorretamente.) Em outras palavras, $A \nsubseteq A \cup B$ sse há um x tal que $x \in A$ e $x \notin A \cup B$. A partir daí, é fácil.

Então, há um $x \in A$ tal que $x \notin A \cup B$. Pela definição de $\cup$, $x \in A \cup B$ sse $x \in A$ ou $x \in B$. Como $x \in A$, temos $x \in A \cup B$. Isso contradiz a suposição de que $x \notin A \cup B$. □

Proposição A.12. *Se $A \subseteq B$ e $B \subseteq C$ então $A \subseteq C$.*

Prova. Suponha $A \subseteq B$ e $B \subseteq C$. Queremos mostrar $A \subseteq C$.

> Vamos proceder indiretamente: supomos a negação do que queremos estabelecer.

Suponha que não, isto é, $A \nsubseteq C$.

> Como antes, raciocinamos que $A \nsubseteq C$ sse nem todo $x \in A$ está também $\in C$, isto é, algum $x \in A$ está $\notin C$. Não se preocupe, com prática você não terá mais que pensar muito para desdobrar negações como esta.

Em outras palavras, há um x tal que $x \in A$ e $x \notin C$.

> Agora podemos usar isso para chegar à nossa contradição. É claro que teremos que usar as outras duas suposições para fazê-lo.

Como $A \subseteq B$, $x \in B$. Como $B \subseteq C$, $x \in C$. Mas isso contradiz $x \notin C$. □

Proposição A.13. *Se $A \cup B = A \cap B$ então $A = B$.*

Prova. Suponha $A \cup B = A \cap B$. Queremos mostrar que $A = B$.

O começo é agora rotineiro:

Suponha, por absurdo, que $A \neq B$.

Nossa suposição para a prova por contradição é que $A \neq B$. Como $A = B$ sse $A \subseteq B$ e $B \subseteq A$, obtemos que $A \neq B$ sse $A \not\subseteq B$ *ou* $B \not\subseteq A$. (Note como é importante ter cuidado ao manipular negações!) Para provar uma contradição a partir dessa disjunção, usamos uma prova por casos e mostramos que, em cada caso, uma contradição se segue.

$A \neq B$ sse $A \not\subseteq B$ ou $B \not\subseteq A$. Distinguimos casos.

No primeiro caso, supomos $A \not\subseteq B$, isto é, para algum x, $x \in A$ mas $\notin B$. $A \cap B$ é definido como aqueles elementos que A e B têm em comum, então, se algo não está em um deles, não está na interseção. $A \cup B$ é A junto com B, então qualquer coisa em um ou outro está também na união. Isso nos diz que $x \in A \cup B$ mas $x \notin A \cap B$, e logo que $A \cap B \neq A \cup B$.

Caso 1: $A \not\subseteq B$. Então, para algum x, $x \in A$ mas $x \notin B$. Como $x \notin B$, então $x \notin A \cap B$. Como $x \in A$, $x \in A \cup B$. Então, $A \cap B \neq A \cup B$, contradizendo a suposição de que $A \cap B = A \cup B$.

Caso 2: $B \not\subseteq A$. Então, para algum y, $y \in B$ mas $y \notin A$. Como antes, temos $y \in A \cup B$ mas $y \notin A \cap B$, e assim $A \cap B \neq A \cup B$, novamente contradizendo $A \cap B = A \cup B$. □

A.8 Lendo Provas

As provas que você encontra em livros-texto e artigos muito raramente fornecem todos os detalhes que incluímos até agora em nossos exemplos. Os autores frequentemente não chamam a atenção para quando distinguem casos, quando dão uma prova indireta, ou não mencionam que usam uma definição. Assim, quando você lê uma prova em um livro-texto, muitas vezes terá de preencher esses detalhes por conta própria para entender a prova. Fazer isso também é um bom exercício para pegar o jeito dos vários passos que você tem de dar em uma prova. Vejamos um exemplo.

Proposição A.14 (Absorção). *Para todos os conjuntos A, B,*

$$A \cap (A \cup B) = A$$

Prova. Se $z \in A \cap (A \cup B)$, então $z \in A$, logo $A \cap (A \cup B) \subseteq A$. Agora suponha $z \in A$. Então também $z \in A \cup B$, e portanto também $z \in A \cap (A \cup B)$. □

A prova precedente da lei da absorção é muito condensada. Não há menção a quaisquer definições usadas, nenhum "temos de provar que" antes de prová-lo, etc. Vamos desempacotá-la. A proposição provada é uma afirmação geral sobre quaisquer conjuntos A e B, e quando a prova menciona A ou B, estes são variáveis para conjuntos arbitrários. As afirmações gerais que a prova estabelece são o que se exige para provar a identidade de conjuntos, isto é, que todo elemento do lado esquerdo da identidade é um elemento do direito e vice-versa.

"Se $z \in A \cap (A \cup B)$, então $z \in A$, logo $A \cap (A \cup B) \subseteq A$."

Esta é a primeira metade da prova da identidade: ela estabelece que, se um z arbitrário é um elemento do lado esquerdo, ele também é um elemento do direito, isto é, $A \cap (A \cup B) \subseteq A$.

Assuma que $z \in A \cap (A \cup B)$. Como z é um elemento da interseção de dois conjuntos se, e somente se, é um elemento de ambos os conjuntos, podemos concluir que $z \in A$ e também $z \in A \cup B$. Em particular, $z \in A$, que é o que queríamos mostrar. Como isso é tudo o que precisa ser feito para a primeira metade, sabemos que o restante da prova deve ser uma prova da segunda metade, isto é, uma prova de que $A \subseteq A \cap (A \cup B)$.

> "Agora suponha $z \in A$. Então também $z \in A \cup B$, e portanto também $z \in A \cap (A \cup B)$."

Começamos assumindo que $z \in A$, já que estamos mostrando que, para qualquer z, se $z \in A$ então $z \in A \cap (A \cup B)$. Para mostrar que $z \in A \cap (A \cup B)$, temos de mostrar (pela definição de "$\cap$") que (i) $z \in A$ e também (ii) $z \in A \cup B$. Aqui (i) é justamente nossa suposição, então não há mais nada a provar, e é por isso que a prova não a menciona novamente. Para (ii), lembre-se de que z é um elemento de uma união de conjuntos se, e somente se, é um elemento de pelo menos um desses conjuntos. Como $z \in A$, e $A \cup B$ é a união de A e B, este é o caso aqui. Então $z \in A \cup B$. Mostramos ambos (i) $z \in A$ e (ii) $z \in A \cup B$, logo, pela definição de "$\cap$," $z \in A \cap (A \cup B)$. A prova não menciona essas definições; presume-se que o leitor já as tenha internalizado. Se você não as internalizou, terá de voltar e relembrar quais são. Então também terá de reconhecer por que decorre de $z \in A$ que $z \in A \cup B$, e de $z \in A$ e $z \in A \cup B$ que $z \in A \cap (A \cup B)$.

Eis outra versão da prova acima, com tudo tornado explícito:

Prova. [Pela definição de $=$ para conjuntos, $A \cap (A \cup B) = A$, temos de mostrar (a) $A \cap (A \cup B) \subseteq A$ e (b) $A \cap (A \cup B) \subseteq A$. (a): Pela definição de $\subseteq$, temos de mostrar que se $z \in A \cap (A \cup B)$, então $z \in A$.] Se $z \in A \cap (A \cup B)$, então $z \in A$ [já que pela definição de $\cap$, $z \in A \cap (A \cup B)$ se, e somente se, $z \in A$ e $z \in A \cup B$], logo $A \cap (A \cup B) \subseteq A$. [(b): Pela definição de $\subseteq$, temos de mostrar que se $z \in A$, então $z \in A \cap (A \cup B)$.] Agora suponha [(1)] $z \in A$.

Então também [(2)] $z \in A \cup B$ [já que por (1) $z \in A$ ou $z \in B$, o que pela definição de $\cup$ significa $z \in A \cup B$], e portanto também $z \in A \cap (A \cup B)$ [já que a definição de $\cap$ requer que $z \in A$, isto é, (1), e $z \in A \cup B$), isto é, (2)]. □

A.9 Não Consigo Fazer Isso!

Todos nós chegamos a um ponto em que sentimos vontade de desistir. Mas você *consegue* fazer isso. Seu professor e o monitor, bem como seus colegas, podem ajudar. Peça ajuda a eles! Aqui estão algumas dicas para ajudá-lo a evitar uma crise, e o que fazer se você sentir vontade de desistir.

Para garantir que você consiga resolver os problemas com sucesso, faça o seguinte:

1. *Comece com a maior antecedência possível.* Ficamos ocupados ao longo do semestre e muitos de nós lutam com a procrastinação; uma das melhores coisas que você pode fazer é começar as tarefas de casa cedo. Dessa forma, se você travar, terá tempo para procurar uma solução (que não seja chorar).

2. *Converse com seus colegas de turma.* Você não está sozinho. Outros na turma também podem ter dificuldades—mas podem ter dificuldades com coisas diferentes. Conversar sobre o assunto com seus colegas pode lhe dar uma perspectiva diferente sobre o problema que pode levar a um avanço. É claro, não copie simplesmente a solução deles: peça uma dica, ou explique onde você travou e peça o próximo passo. E quando você conseguir, retribua. Ajudar outra pessoa, e explicar as coisas, ajudará você a entender melhor também.

3. *Peça ajuda.* Você tem muitos recursos à sua disposição—seu professor e o monitor estão lá por você e *querem* que você

tenha sucesso. Eles devem ser capazes de ajudá-lo a resolver um problema e identificar em que ponto do processo você está com dificuldade.

4. *Faça uma pausa.* Se você travar, *pode* ser porque ficou olhando para o problema por tempo demais. Faça uma pausa curta, tome uma xícara de chá, ou trabalhe em um problema diferente por um tempo, depois volte ao problema com a mente renovada. Durma sobre o assunto.

Percebe como essas estratégias exigem que você tenha começado a trabalhar na prova com bastante antecedência? Se você começou a prova às 2 da madrugada do dia anterior à entrega, elas talvez não sejam tão úteis.

Isso pode soar como pessimismo e desgraça, mas resolver uma prova é um desafio que compensa no final. Algumas pessoas fazem isso como profissão—portanto deve haver algo de prazeroso nisso. Como praticamente tudo, resolver problemas e fazer provas é algo que exige prática. Você pode ver colegas que acham isso fácil: provavelmente eles apenas já tiveram muita prática. Tente não ceder com facilidade demais.

Se você ficar mesmo sem tempo (ou sem paciência) em um problema específico: tudo bem. Isso não significa que você seja burro ou que nunca vá conseguir. Descubra (com seu professor ou outro estudante) como se faz, e identifique onde você errou ou travou, para poder evitar fazer isso da próxima vez que encontrar uma questão semelhante. Depois tente fazê-lo sem olhar a solução. E na próxima vez, comece (e peça ajuda) mais cedo.

A.10 Outros Recursos

Há muitos livros sobre como fazer provas em matemática que podem ser úteis. Confira em particular *How to Read and do Proofs: An Introduction to Mathematical Thought Processes* (Solow, 2013) e *How to Prove It: A Structured Approach* (Velleman, 2019). O *Book of Proof* (Hammack, 2013) e o *Mathematical Reasoning* (Sandstrum,

2019) são livros sobre provas que estão disponíveis gratuitamente online. Filósofos podem achar *More Precisely: The Math you need to do Philosophy* (Steinhart, 2018) uma boa introdução ao raciocínio matemático.

Há também vários guias mais curtos sobre provas disponíveis na internet; por exemplo, "Introduction to Mathematical Arguments" (Hutchings, 2003) e "How to write proofs" (Cheng, 2004).

Vídeos Motivacionais

Sente que não tem motivação para fazer sua tarefa de casa? Desanimado? Estes vídeos podem ajudar!

- https://www.youtube.com/watch?v=ZXsQAXx_ao0
- https://www.youtube.com/watch?v=BQ4yd2W50No
- https://www.youtube.com/watch?v=StTqXEQ2l-Y

Problemas

Problema A.1. Suponha que lhe peçam para provar que $A \cap B \neq \emptyset$. Desdobre todas as definições que ocorrem aqui, isto é, reenuncie isto de um modo que não mencione "$\cap$", "=", ou "$\emptyset$".

Problema A.2. Prove *indiretamente* que $A \cap B \subseteq A$.

Problema A.3. Expanda a seguinte prova de $A \cup (A \cap B) = A$, mencionando todos os padrões de inferência usados, por que cada passo decorre de suposições ou afirmações estabelecidas antes dele, e onde temos de apelar para quais definições.

Prova. Se $z \in A \cup (A \cap B)$ então $z \in A$ ou $z \in A \cap B$. Se $z \in A \cap B$, $z \in A$. Qualquer $z \in A$ está também $\in A \cup (A \cap B)$. □

APÊNDICE B

Indução

B.1 Introdução

A indução é uma técnica de prova importante, usada, em diferentes formas, em quase todas as áreas da lógica, da ciência da computação teórica e da matemática. Ela é necessária para provar muitos dos resultados em lógica.

A indução é frequentemente contrastada com a dedução, e caracterizada como a inferência do particular para o geral. Por exemplo, se observamos muitas esmeraldas verdes, e nada que chamaríamos de esmeralda que não seja verde, poderíamos concluir que todas as esmeraldas são verdes. Essa é uma inferência indutiva, no sentido de que procede de muitos casos particulares (esta esmeralda é verde, aquela esmeralda é verde, etc.) a uma afirmação geral (todas as esmeraldas são verdes). A indução *matemática* também é uma inferência que conclui uma afirmação geral, mas é de um tipo muito diferente dessa "indução simples."

De modo bem aproximado, uma prova indutiva em matemática conclui que todos os objetos matemáticos de certo tipo têm uma certa propriedade. No caso mais simples, os objetos matemáticos com que uma prova indutiva se ocupa são os números naturais. Nesse caso, uma prova indutiva é usada para estabelecer que todos os números naturais têm alguma propriedade, e faz isso mostrando que

1. 0 tem a propriedade, e

2. sempre que um número k tem a propriedade, também $k+1$ a tem.

A indução sobre os números naturais pode então, também, ser frequentemente usada para provar afirmações gerais sobre objetos matemáticos aos quais se podem atribuir números. Por exemplo, conjuntos finitos têm, cada um, um número finito n de elementos, e se pudermos usar a indução para mostrar que todo número n tem a propriedade "todos os conjuntos finitos de tamanho n são ..." então teremos mostrado algo sobre todos os conjuntos finitos.

A indução também pode ser generalizada para objetos matemáticos que são *definidos indutivamente.* Por exemplo, as expressões de uma linguagem formal, como as da lógica de primeira ordem, são definidas indutivamente. A *indução estrutural* é um modo de provar resultados sobre todas essas expressões. A indução estrutural, em particular, é muito útil—e amplamente usada—em lógica.

B.2 Indução em $\mathbb{N}$

Em sua forma mais simples, a indução é uma técnica usada para provar resultados para todos os números naturais. Ela usa o fato de que, começando em 0 e somando 1 repetidamente, eventualmente alcançamos todo número natural. Assim, para provar que algo é verdadeiro para todo número, podemos (1) estabelecer que é verdadeiro para 0 e (2) mostrar que, sempre que é verdadeiro para um número n, também é verdadeiro para o número seguinte $n+1$. Se abreviarmos "o número n tem a propriedade P" por $P(n)$ (e "o número k tem a propriedade P" por $P(k)$, etc.), então uma prova por indução de que $P(n)$ para todo $n \in \mathbb{N}$ consiste em:

1. uma prova de $P(0)$, e

2. uma prova de que, para qualquer k, se $P(k)$ então $P(k+1)$.

Para deixar isso bem claro, suponha que tenhamos tanto (1) quanto (2). Então (1) nos diz que $P(0)$ é verdadeiro. Se também tivermos (2), sabemos em particular que, se $P(0)$, então $P(0+1)$, isto é, $P(1)$. Isso decorre da afirmação geral "para qualquer k, se $P(k)$ então $P(k+1)$" ao pôr 0 no lugar de k. Então, por modus ponens, temos que $P(1)$. De (2) novamente, agora tomando 1 por n, temos: se $P(1)$ então $P(2)$. Como acabamos de estabelecer $P(1)$, por modus ponens, temos $P(2)$. E assim por diante. Para qualquer número n, depois de fazer isso n vezes, eventualmente chegamos a $P(n)$. Então (1) e (2) juntos estabelecem $P(n)$ para qualquer $n \in \mathbb{N}$.

Vejamos um exemplo. Suponha que queiramos descobrir quantas somas diferentes podemos lançar com n dados. Embora possa parecer bobo, vamos começar com 0 dados. Se você não tem dados, há apenas uma soma possível que você pode "lançar": nenhum ponto, que soma 0. Assim, o número de lançamentos possíveis diferentes é 1. Se você tem apenas um dado, isto é, $n = 1$, há seis valores possíveis, de 1 a 6. Com dois dados, podemos lançar qualquer soma de 2 a 12, ou seja, 11 possibilidades. Com três dados, podemos lançar qualquer número de 3 a 18, isto é, 16 possibilidades diferentes. 1, 6, 11, 16: parece um padrão: talvez a resposta seja $5n + 1$? É claro que $5n + 1$ é o máximo possível, pois há apenas $5n + 1$ números entre n, o menor valor que você pode lançar com n dados (todos 1) e $6n$, o maior que você pode lançar (todos 6).

Teorema B.1. *Com n dados pode-se lançar todos os $5n + 1$ valores possíveis entre n e $6n$.*

Prova. Seja $P(n)$ a afirmação: "É possível lançar qualquer número entre n e $6n$ usando n dados." Para usar a indução, provamos:

1. A *base da indução* $P(1)$, isto é, com apenas um dado, você pode lançar qualquer número entre 1 e 6.

2. O *passo da indução*, para todo k, se $P(k)$ então $P(k+1)$.

(1) é provado inspecionando um dado de 6 lados. Ele tem todos os 6 lados, e todo número entre 1 e 6 aparece em um dos lados. Então é possível lançar qualquer número entre 1 e 6 usando um único dado.

Para provar (2), supomos o antecedente do condicional, isto é, $P(k)$. Essa suposição é chamada de *hipótese indutiva*. Nós a usamos para provar $P(k+1)$. A parte difícil é encontrar um modo de pensar sobre os valores possíveis de um lançamento de $k+1$ dados em termos dos valores possíveis de lançamentos de k dados, somados aos lançamentos do dado extra, o $(k+1)$-ésimo—é isso, porém, que temos que fazer, se quisermos usar a hipótese indutiva.

A hipótese indutiva diz que podemos obter qualquer número entre k e $6k$ usando k dados. Se lançamos um 1 com nosso $(k+1)$-ésimo dado, isso acrescenta 1 ao total. Então podemos lançar qualquer valor entre $k+1$ e $6k+1$ lançando k dados e depois rolando um 1 com o $(k+1)$-ésimo dado. O que falta? Os valores de $6k+2$ a $6k+6$. Podemos obtê-los rolando k dados em 6 e depois um número entre 2 e 6 com nosso $(k+1)$-ésimo dado. Juntos, isso significa que, com $k+1$ dados, podemos lançar qualquer um dos números entre $k+1$ e $6(k+1)$, isto é, provamos $P(k+1)$ usando a suposição $P(k)$, a hipótese indutiva. □

Muito frequentemente usamos a indução quando queremos provar algo sobre uma série de objetos (números, conjuntos, etc.) que é, ela própria, definida "indutivamente," isto é, definindo o $(n+1)$-ésimo objeto em termos do n-ésimo. Por exemplo, podemos definir a soma s_n dos números naturais até n por

$$s_0 = 0$$
$$s_{n+1} = s_n + (n+1)$$

Esta definição dá:

$$\begin{aligned} s_0 &= 0, \\ s_1 &= s_0 + 1 && = 1, \\ s_2 &= s_1 + 2 && = 1 + 2 = 3 \\ s_3 &= s_2 + 3 && = 1 + 2 + 3 = 6, \text{ etc.} \end{aligned}$$

Agora podemos provar, por indução, que $s_n = n(n+1)/2$.

Proposição B.2. $s_n = n(n+1)/2$.

Prova. Temos que provar (1) que $s_0 = 0 \cdot (0+1)/2$ e (2) se $s_k = k(k+1)/2$ então $s_{k+1} = (k+1)(k+2)/2$. (1) é óbvio. Para provar (2), supomos a hipótese indutiva: $s_k = k(k+1)/2$. Usando-a, temos que mostrar que $s_{k+1} = (k+1)(k+2)/2$.

O que é s_{k+1}? Pela definição, $s_{k+1} = s_k + (k+1)$. Pela hipótese indutiva, $s_k = k(k+1)/2$. Podemos substituir isso na equação anterior, e então só precisamos de um pouco de aritmética de frações:

$$\begin{aligned} s_{k+1} &= \frac{k(k+1)}{2} + (k+1) = \\ &= \frac{k(k+1)}{2} + \frac{2(k+1)}{2} = \\ &= \frac{k(k+1) + 2(k+1)}{2} = \\ &= \frac{(k+2)(k+1)}{2}. \end{aligned}$$

□

A lição importante aqui é que, se você está provando algo sobre alguma sequência definida indutivamente a_n, a indução é o caminho óbvio a seguir. E, mesmo quando não é (como no caso das possibilidades de lançamentos de dados), você pode usar a indução se conseguir, de alguma forma, relacionar o caso para $k+1$ ao caso para k.

B.3 Indução Forte

No princípio de indução discutido acima, provamos $P(0)$ e também que, se $P(k)$, então $P(k+1)$. Na segunda parte, assumimos que $P(k)$ é verdadeiro e usamos essa suposição para provar $P(k+1)$. De forma equivalente, é claro, poderíamos assumir $P(k-1)$ e usá-la para provar $P(k)$—o importante é que sejamos capazes de realizar a inferência de qualquer número para seu sucessor; que possamos provar a afirmação em questão para qualquer número sob a suposição de que ela vale para seu antecessor.

Há uma variante do princípio de indução em que não apenas assumimos que a afirmação vale para o antecessor $k-1$ de k, mas para todos os números menores que k, e usamos essa suposição para estabelecer a afirmação para k. Isso também nos dá a afirmação $P(n)$ para todo $n \in \mathbb{N}$. Pois, uma vez que tenhamos estabelecido $P(0)$, teremos com isso estabelecido que P vale para todos os números menores que 1. E se sabemos que, se $P(l)$ para todo $l < k$, então $P(k)$, sabemos isso em particular para $k = 1$. Então podemos concluir $P(1)$. Com isso provamos $P(0)$ e $P(1)$, isto é, $P(l)$ para todo $l < 2$, e como temos também o condicional, se $P(l)$ para todo $l < 2$, então $P(2)$, podemos concluir $P(2)$, e assim por diante.

De fato, se conseguirmos estabelecer o condicional geral "para todo k, se $P(l)$ para todo $l < k$, então $P(k)$," não temos mais de estabelecer $P(0)$, já que ele decorre disso. Pois lembre-se de que uma afirmação geral como "para todo $l < k$, $P(l)$" é verdadeira se não houver nenhum $l < k$. Este é um caso de quantificação vazia: "todos os As são Bs" é verdadeiro se não houver nenhum A, $\forall x\,(A(x) \to B(x))$ é verdadeiro se nenhum x satisfaz $A(x)$. Neste caso, a versão formalizada seria "$\forall l\,(l < k \to P(l))$"—e isso é verdadeiro se não houver nenhum $l < k$. E se $k = 0$ esse é exatamente o caso: não há nenhum $l < 0$, logo "para todo $l < 0$, $P(0)$" é verdadeiro, qualquer que seja P. Uma prova de "se $P(l)$ para todo $l < k$, então $P(k)$" estabelece assim automaticamente $P(0)$.

Esta variante é útil se estabelecer a afirmação para k não puder ser feito de modo a apenas se apoiar na afirmação para $k-1$, mas puder exigir a suposição de que ela é verdadeira para um ou mais $l < k$.

B.4 Definições Indutivas

Em lógica, muito frequentemente definimos tipos de objetos *indutivamente*, isto é, especificando regras para o que conta como um objeto do tipo a ser definido, que explicam como obter novos objetos desse tipo a partir de objetos antigos desse tipo. Por exemplo, frequentemente definimos tipos especiais de sequências de símbolos, como os termos e fórmulas de uma linguagem, por indução. Para um exemplo simples, considere cadeias compostas das letras a, b, c, d, do símbolo ∘, e dos colchetes [e], como "[[c ∘ d][", "[a[]∘]", "a" ou "[[a ∘ b] ∘ d]". Você provavelmente sente que há algo "errado" com as duas primeiras cadeias: os colchetes não "se equilibram" de modo algum na primeira, e você pode sentir que o "∘" deveria "conectar" expressões que elas próprias façam sentido. A terceira e a quarta cadeia parecem melhores: para todo "[" há um "]" de fechamento (se houver algum), e para qualquer ∘ podemos encontrar expressões "bonitas" de cada lado, cercadas por um par de parênteses.

Gostaríamos de especificar precisamente o que conta como um "termo bonito." Em primeiro lugar, toda letra isolada é bonita. Qualquer coisa que não seja apenas uma letra isolada deve ser da forma "[$t \circ s$]" em que s e t são, eles próprios, bonitos. Reciprocamente, se t e s são bonitos, então podemos formar um novo termo bonito pondo um ∘ entre eles e cercando-os por um par de colchetes. Poderíamos usar essas operações para *definir* o conjunto dos termos bonitos. Esta é uma *definição indutiva.*

Definição B.3 (Termos bonitos). O conjunto dos *termos bonitos* é definido indutivamente como segue:

1. Qualquer letra a, b, c, d é um termo bonito.
2. Se s_1 e s_2 são termos bonitos, então também o é $[s_1 \circ s_2]$.
3. Nada mais é um termo bonito.

Esta definição nos diz que algo conta como um termo bonito sse ele pode ser construído de acordo com as duas condições (1) e (2) em algum número finito de passos. No primeiro passo, construímos todos os termos bonitos compostos apenas por letras isoladas, isto é,

$$\mathrm{a}, \mathrm{b}, \mathrm{c}, \mathrm{d}$$

No segundo passo, aplicamos (2) aos termos que construímos. Obteremos

$$[\mathrm{a} \circ \mathrm{a}], [\mathrm{a} \circ \mathrm{b}], [\mathrm{b} \circ \mathrm{a}], \ldots, [\mathrm{d} \circ \mathrm{d}]$$

para todas as combinações de duas letras. No terceiro passo, aplicamos (2) novamente, a quaisquer dois termos bonitos que construímos até agora. Obtemos novos termos bonitos como $[\mathrm{a} \circ [\mathrm{a} \circ \mathrm{a}]]$—em que t é a do passo 1 e s é $[\mathrm{a} \circ \mathrm{a}]$ do passo 2—e $[[\mathrm{b} \circ \mathrm{c}] \circ [\mathrm{d} \circ \mathrm{b}]]$ construído a partir dos dois termos $[\mathrm{b} \circ \mathrm{c}]$ e $[\mathrm{d} \circ \mathrm{b}]$ do passo 2. E assim por diante. A cláusula (3) impede que qualquer coisa não construída desse modo se infiltre no conjunto dos termos bonitos.

Note que ainda não provamos que toda sequência de símbolos que "parece" bonita é bonita de acordo com esta definição. No entanto, deve ficar claro que tudo o que podemos construir de fato "parece bonito": os colchetes estão equilibrados, e $\circ$ conecta partes que são, elas próprias, bonitas.

A característica-chave das definições indutivas é que, se você quer provar algo sobre todos os termos bonitos, a definição lhe diz quais casos você deve considerar. Por exemplo, se lhe é dito que t é um termo bonito, a definição indutiva lhe diz com o que t pode se parecer: t pode ser uma letra, ou pode ser $[s_1 \circ s_2]$ para algum par de termos bonitos s_1 e s_2. Por causa da cláusula (3), essas são as únicas possibilidades.

Ao provar afirmações sobre todo um conjunto definido indutivamente, a forma forte da indução torna-se particularmente importante. Por exemplo, suponha que queiramos provar que, para todo termo bonito de comprimento n, o número de [nele é $< n/2$. Isso pode ser visto como uma afirmação sobre todos os n: para todo n, o número de [em qualquer termo bonito de comprimento n é $< n/2$.

Proposição B.4. *Para qualquer n, o número de [em um termo bonito de comprimento n é $< n/2$.*

Prova. Para provar esse resultado por indução (forte), temos que mostrar que a seguinte afirmação condicional é verdadeira:

> Se, para todo $l < k$, qualquer termo bonito de comprimento l tem $< l/2$ [, então qualquer termo bonito de comprimento k tem $< k/2$ [.

Para mostrar este condicional, suponha que seu antecedente é verdadeiro, isto é, suponha que, para qualquer $l < k$, termos bonitos de comprimento l contêm $< l/2$ [. Chamamos essa suposição de hipótese indutiva. Queremos mostrar que o mesmo vale para termos bonitos de comprimento k.

Então suponha que t é um termo bonito de comprimento k. Como os termos bonitos são definidos indutivamente, temos dois casos: (1) t é uma letra isolada, ou (2) t é $[s_1 \circ s_2]$ para alguns termos bonitos s_1 e s_2.

1. t é uma letra. Então $k = 1$, e o número de [em t é 0. Como $0 < 1/2$, a afirmação vale.

2. t é $[s_1 \circ s_2]$ para alguns termos bonitos s_1 e s_2. Sejam l_1 o comprimento de s_1 e l_2 o comprimento de s_2. Então o comprimento k de t é $l_1 + l_2 + 3$ (os comprimentos de s_1 e s_2 mais três símbolos [, $\circ$,]). Como $l_1 + l_2 + 3$ é sempre maior que l_1, $l_1 < k$. De modo semelhante, $l_2 < k$. Isso significa que a hipótese de indução se aplica aos termos s_1

e s_2: o número m_1 de [em s_1 é $< l_1/2$, e o número m_2 de [em s_2 é $< l_2/2$.

O número de [em t é o número de [em s_1, mais o número de [em s_2, mais 1, isto é, é $m_1 + m_2 + 1$. Como $m_1 < l_1/2$ e $m_2 < l_2/2$, temos:

$$m_1 + m_2 + 1 < \frac{l_1}{2} + \frac{l_2}{2} + 1 = \frac{l_1 + l_2 + 2}{2} < \frac{l_1 + l_2 + 3}{2} = k/2.$$

Em cada caso, mostramos que o número de [em t é $< k/2$ (com base na hipótese indutiva). Por indução forte, a proposição se segue. □

B.5 Indução Estrutural

Até aqui usamos a indução para estabelecer resultados sobre todos os números naturais. Mas um princípio correspondente pode ser usado diretamente para provar resultados sobre todos os elementos de um conjunto definido indutivamente. Isso é frequentemente chamado de indução *estrutural*, porque depende da estrutura dos objetos definidos indutivamente.

Em geral, uma definição indutiva é dada por (a) uma lista de elementos "iniciais" do conjunto e (b) uma lista de operações que produzem novos elementos do conjunto a partir de antigos. No caso dos termos bonitos, por exemplo, os objetos iniciais são as letras. Temos apenas uma operação: as operações são

$$o(s_1, s_2) = [s_1 \circ s_2]$$

Você pode até pensar nos próprios números naturais $\mathbb{N}$ como sendo dados por uma definição indutiva: o objeto inicial é 0, e a operação é a função sucessor $x + 1$.

Para provar algo sobre todos os elementos de um conjunto definido indutivamente, isto é, que todo elemento do conjunto tem uma propriedade P, devemos:

1. Provar que os objetos iniciais têm P

2. Provar que, para cada operação o, se os argumentos têm P, o resultado também tem.

Por exemplo, para provar algo sobre todos os termos bonitos, nós provaríamos que é verdadeiro para todas as letras, e que é verdadeiro para $[s_1 \circ s_2]$ desde que seja verdadeiro de s_1 e s_2 individualmente.

Proposição B.5. *O número de* [*é igual ao número de*] *em qualquer termo bonito t.*

Prova. Usamos a indução estrutural. Os termos bonitos são definidos indutivamente, com as letras como objetos iniciais e a operação o para construir novos termos bonitos a partir de antigos.

1. A afirmação é verdadeira para toda letra, já que o número de [em uma letra isolada é 0 e o número de] nela também é 0.

2. Suponha que o número de [em s_1 é igual ao número de], e o mesmo é verdadeiro para s_2. O número de [em $o(s_1, s_2)$, isto é, em $[s_1 \circ s_2]$, é a soma do número de [em s_1 e s_2 mais um. O número de] em $o(s_1, s_2)$ é a soma do número de] em s_1 e s_2 mais um. Assim, o número de [em $o(s_1, s_2)$ é igual ao número de] em $o(s_1, s_2)$. □

Vamos dar outra prova por indução estrutural: um segmento inicial próprio de uma cadeia t de símbolos é qualquer cadeia s que concorda com t símbolo por símbolo, lida da esquerda, mas t é mais longa. Assim, por exemplo, $[a \circ$ é um segmento inicial próprio de $[a \circ b]$, mas nem $[b \circ$ (discordam no segundo símbolo) nem $[a \circ b]$ (têm o mesmo comprimento) o são.

Proposição B.6. *Todo segmento inicial próprio de um termo bonito t tem mais* [*do que*].

Prova. Por indução em t:

1. t é uma letra isolada: Então t não tem segmentos iniciais próprios.

2. $t = [s_1 \circ s_2]$ para alguns termos bonitos s_1 e s_2. Se r é um segmento inicial próprio de t, há várias possibilidades:

   a) r é apenas [: Então r tem um [a mais do que].

   b) r é $[r_1$ em que r_1 é um segmento inicial próprio de s_1: Como s_1 é um termo bonito, por hipótese de indução, r_1 tem mais [do que], e o mesmo é verdadeiro para $[r_1$.

   c) r é $[s_1$ ou $[s_1 \circ$: Pelo resultado anterior, o número de [e] em s_1 é igual; então o número de [em $[s_1$ ou $[s_1 \circ$ é um a mais do que o número de].

   d) r é $[s_1 \circ r_2$ em que r_2 é um segmento inicial próprio de s_2: Por hipótese de indução, r_2 contém mais [do que]. Pelo resultado anterior, o número de [e de] em s_1 é igual. Então o número de [em $[s_1 \circ r_2$ é maior do que o número de].

   e) r é $[s_1 \circ s_2$: Pelo resultado anterior, o número de [e] em s_1 é igual, e o mesmo para s_2. Então há um [a mais em $[s_1 \circ s_2$ do que há]. □

B.6 Relações e Funções

Quando definimos um conjunto de objetos (como os números naturais ou os termos bonitos) indutivamente, também podemos definir *relações sobre* esses objetos por indução. Por exemplo, considere a seguinte ideia: um termo bonito t_1 é um subtermo de um termo bonito t_2 se ocorre como parte dele. Vamos usar um símbolo para isso: $t_1 \sqsubseteq t_2$. Todo termo bonito é, é claro, um subtermo de si mesmo: $t \sqsubseteq t$. Podemos dar uma definição indutiva dessa relação como segue:

Definição B.7. A relação de um termo bonito t_1 ser um subtermo de t_2, $t_1 \sqsubseteq t_2$, é definida por indução em t_2 como segue:

1. Se t_2 é uma letra, então $t_1 \sqsubseteq t_2$ sse $t_1 = t_2$.
2. Se t_2 é $[s_1 \circ s_2]$, então $t_1 \sqsubseteq t_2$ sse $t_1 = t_2$, $t_1 \sqsubseteq s_1$, ou $t_1 \sqsubseteq s_2$.

Esta definição, por exemplo, nos dirá que a $\sqsubseteq$ [b $\circ$ a]. Pois (2) diz que a $\sqsubseteq$ [b $\circ$ a] sse a = [b $\circ$ a], ou a $\sqsubseteq$ *b*, ou a $\sqsubseteq$ a. As duas primeiras são falsas: a claramente não é idêntico a [b $\circ$ a], e por (1), a $\sqsubseteq$ b sse a = b, o que também é falso. No entanto, também por (1), a $\sqsubseteq$ a sse a = a, o que é verdadeiro.

É importante observar que o sucesso desta definição depende de um fato que ainda não provamos: todo termo bonito t ou é uma letra isolada, ou existem termos bonitos *univocamente determinados* s_1 e s_2 tais que $t = [s_1 \circ s_2]$. "Univocamente determinados" aqui significa que, se $t = [s_1 \circ s_2]$, ele não é *também* $= [r_1 \circ r_2]$ com $s_1 \neq r_1$ ou $s_2 \neq r_2$. Se este fosse o caso, então a cláusula (2) poderia entrar em conflito consigo mesma: lendo t_2 como $[s_1 \circ s_2]$ poderíamos obter $t_1 \sqsubseteq t_2$, mas se lermos t_2 como $[r_1 \circ r_2]$ poderíamos obter não $t_1 \sqsubseteq t_2$. Antes de provarmos que isso não pode acontecer, vejamos um exemplo em que ele *pode* acontecer.

Definição B.8. Defina *termos sem colchetes* indutivamente por

1. Toda letra é um termo sem colchetes.
2. Se s_1 e s_2 são termos sem colchetes, então $s_1 \circ s_2$ é um termo sem colchetes.
3. Nada mais é um termo sem colchetes.

Termos sem colchetes são, por exemplo, a, b $\circ$ d, b $\circ$ a $\circ$ b. Agora, se definíssemos "subtermo" para termos sem colchetes do modo como fizemos acima, a segunda cláusula diria

Se $t_2 = s_1 \circ s_2$, então $t_1 \sqsubseteq t_2$ sse $t_1 = t_2$, $t_1 \sqsubseteq s_1$, ou $t_1 \sqsubseteq s_2$.

Agora, b ∘ a ∘ b é da forma $s_1 \circ s_2$ com

$$s_1 = \text{b e} \qquad\qquad s_2 = \text{a} \circ \text{b}.$$

Ele também é da forma $r_1 \circ r_2$ com

$$r_1 = \text{b} \circ \text{a e} \qquad\qquad r_2 = \text{b}.$$

Agora, a ∘ b é um subtermo de b ∘ a ∘ b? A resposta é sim, se seguirmos a primeira leitura, e não, se seguirmos a segunda.

A propriedade de que o modo como um termo bonito é construído a partir de outros termos bonitos é único é chamada de *leitura única*. Como definições indutivas de relações para tais objetos definidos indutivamente são importantes, temos que provar que ela vale.

Proposição B.9. *Suponha que t é um termo bonito. Então ou t é uma letra isolada, ou existem termos bonitos univocamente determinados s_1, s_2 tais que $t = [s_1 \circ s_2]$.*

Prova. Se t é uma letra isolada, a condição é satisfeita. Então suponha que t não é uma letra isolada. Podemos perceber, pela definição indutiva, que então t deve ser da forma $[s_1 \circ s_2]$ para alguns termos bonitos s_1 e s_2. Resta mostrar que estes são univocamente determinados, isto é, se $t = [r_1 \circ r_2]$, então $s_1 = r_1$ e $s_2 = r_2$.

Então suponha $t = [s_1 \circ s_2]$ e também $t = [r_1 \circ r_2]$ para termos bonitos s_1, s_2, r_1, r_2. Temos que mostrar que $s_1 = r_1$ e $s_2 = r_2$. Primeiro, s_1 e r_1 devem ser idênticos, pois, caso contrário, um é um segmento inicial próprio do outro. Mas, por Proposição B.6, isso é impossível se s_1 e r_1 são ambos termos bonitos. Mas se $s_1 = r_1$, então claramente também $s_2 = r_2$. □

Também podemos definir funções indutivamente: por exemplo, podemos definir a função f que mapeia qualquer termo bonito na profundidade máxima de aninhamento de $[\ldots]$ nele como segue:

Definição B.10. A *profundidade* de um termo bonito, $f(t)$, é definida indutivamente como segue:

$$f(t) = \begin{cases} 0 & \text{se } t \text{ é uma letra} \\ \max(f(s_1), f(s_2)) + 1 & \text{se } t = [s_1 \circ s_2]. \end{cases}$$

Por exemplo,

$$\begin{aligned} f([\mathrm{a} \circ \mathrm{b}]) &= \max(f(\mathrm{a}), f(\mathrm{b})) + 1 = \\ &= \max(0,0) + 1 = 1, \text{ e} \\ f([[\mathrm{a} \circ \mathrm{b}] \circ \mathrm{c}]) &= \max(f([\mathrm{a} \circ \mathrm{b}]), f(\mathrm{c})) + 1 = \\ &= \max(1,0) + 1 = 2. \end{aligned}$$

Aqui, é claro, supomos que s_1 e s_2 são termos bonitos, e fazemos uso do fato de que todo termo bonito ou é uma letra ou é da forma $[s_1 \circ s_2]$. É novamente importante que ele possa ser dessa forma de apenas um modo. Para ver por quê, considere novamente os termos sem colchetes que definimos antes. A "definição" correspondente seria:

$$g(t) = \begin{cases} 0 & \text{se } t \text{ é uma letra} \\ \max(g(s_1), g(s_2)) + 1 & \text{se } t = s_1 \circ s_2. \end{cases}$$

Agora considere o termo sem colchetes $\mathrm{a} \circ \mathrm{b} \circ \mathrm{c} \circ \mathrm{d}$. Ele pode ser lido de mais de um modo, por exemplo, como $s_1 \circ s_2$ com

$$s_1 = \mathrm{a} \text{ e} \qquad s_2 = \mathrm{b} \circ \mathrm{c} \circ \mathrm{d},$$

ou como $r_1 \circ r_2$ com

$$r_1 = \mathrm{a} \circ b \text{ e} \qquad r_2 = \mathrm{c} \circ \mathrm{d}.$$

Calcular g de acordo com o primeiro modo de lê-lo daria

$$\begin{aligned} g(s_1 \circ s_2) &= \max(g(\mathrm{a}), g(\mathrm{b} \circ \mathrm{c} \circ \mathrm{d})) + 1 = \\ &= \max(0,2) + 1 = 3 \end{aligned}$$

enquanto, de acordo com a outra leitura, obtemos

$$\begin{aligned} g(r_1 \circ r_2) &= \max(g(\mathrm{a} \circ \mathrm{b}), g(\mathrm{c} \circ \mathrm{d})) + 1 = \\ &= \max(1,1) + 1 = 2 \end{aligned}$$

Mas uma função deve sempre produzir um valor único; então nossa "definição" de g não define função alguma.

Problemas

Problema B.1. Defina o conjunto dos termos superbonitos por

1. Qualquer letra a, b, c, d é um termo superbonito.
2. Se s é um termo superbonito, então também o é $[s]$.
3. Se s_1 e s_2 são termos superbonitos, então também o é $[s_1 \circ s_2]$.
4. Nada mais é um termo superbonito.

Mostre que o número de [em um termo superbonito t de comprimento n é $\leq n/2 + 1$.

Problema B.2. Prove, por indução estrutural, que nenhum termo bonito começa com].

Problema B.3. Dê uma definição indutiva da função l, em que $l(t)$ é o número de símbolos no termo bonito t.

Problema B.4. Prove, por indução estrutural sobre termos bonitos t, que $f(t) < l(t)$ (em que $l(t)$ é o número de símbolos em t e $f(t)$ é a profundidade de t tal como definida em Definição B.10).

APÊNDICE C

Biografias

C.1 Georg Cantor

Uma biografia inicial de Georg Cantor (GAY-org KAHN-tor) afirmava que ele tinha nascido e sido encontrado num navio que seguia para São Petersburgo, na Rússia, e que seus pais eram desconhecidos. Isto, porém, não é verdade; embora ele tivesse nascido em São Petersburgo em 1845.

Fig. C.1: Georg Cantor

Cantor se doutorou em matemática na Universidade de Berlim em 1867. É conhecido por seu trabalho em teoria dos conjuntos e é considerado o fundador da teoria dos conjuntos como disciplina de pesquisa distinta. Foi o primeiro a provar que existem conjuntos infinitos de tamanhos diferentes. Suas teorias, e sobretudo a teoria dos infinitos, provoca-

ram muito debate entre os matemáticos da época, e seu trabalho foi controverso.

As convicções religiosas de Cantor e seu trabalho matemático estavam inextricavelmente ligados; ele chegou a afirmar que a teoria dos números transfinitos lhe tinha sido comunicada diretamente por Deus. Mais tarde, Cantor sofreu de doença mental. A partir de 1894, e com maior frequência nos últimos anos, Cantor foi internado. As fortes críticas a seu trabalho, incluindo um rompimento com o matemático Leopold Kronecker, levaram à depressão e à perda de interesse pela matemática. Durante episódios depressivos, Cantor se dedicava à filosofia e à literatura, e chegou a publicar uma teoria de que Francis Bacon era o autor das peças de Shakespeare.

Cantor morreu em 6 de janeiro de 1918, num sanatório em Halle.

Leitura complementar Para biografias completas de Cantor, ver Dauben (1990) e Grattan-Guinness (1971). As visões radicais de Cantor são também descritas no programa da BBC Radio 4 *A Brief History of Mathematics* (du Sautoy, 2014). Se quiser ouvir as teorias de Cantor em forma de rap, ver Rose (2012).

C.2 Alonzo Church

Alonzo Church nasceu em Washington, DC, em 14 de junho de 1903. Na primeira infância, um acidente com uma arma de ar comprimido deixou Church cego de um olho. Ele concluiu o ensino preparatório em Connecticut em 1920 e iniciou sua formação universitária em Princeton naquele mesmo ano. Concluiu seus estudos de doutorado em 1927. Após alguns anos no exterior, Church retornou a Princeton. Church era conhecido por ser extremamente educado e cuidadoso. Sua escrita no quadro-negro era impecável, e ele preservava artigos importantes cobrindo-os cuidadosamente com cimento Duco (uma cola transparente). Fora de suas atividades acadêmicas, gostava de ler revistas de

ficção científica e não tinha receio de escrever aos editores caso identificasse quaisquer imprecisões nos textos.

Fig. C.2: Alonzo Church

As realizações acadêmicas de Church foram notáveis. Junto com seus alunos Stephen Kleene e Barkley Rosser, ele desenvolveu uma teoria da calculabilidade efetiva, o cálculo lambda, independentemente do desenvolvimento da máquina de Turing por Alan Turing. As duas definições de computabilidade são equivalentes e dão origem ao que hoje é conhecido como a *Tese de Church–Turing*: a de que uma função dos números naturais é efetivamente computável se e somente se for computável por meio de uma máquina de Turing (ou do cálculo lambda). Ele também provou o que hoje é conhecido como o *Teorema de Church*: o problema da decisão para a validade de fórmulas de primeira ordem é insolúvel.

Church continuou seu trabalho até a velhice. Em 1967, deixou Princeton para ir à UCLA, onde foi professor até se aposentar, em 1990. Church faleceu em 1º de agosto de 1995, aos 92 anos.

Leitura complementar Para uma breve biografia de Church, veja Enderton (2019). Os escritos originais de Church sobre o cálculo lambda e o Entscheidungsproblem (Tese de Church) são Church (1936a,b). Aspray (1984) registra uma entrevista com Church sobre a comunidade matemática de Princeton na década de 1930. Church escreveu uma série de resenhas de livros para o

Journal of Symbolic Logic de 1936 até 1979. Todas estão arquivadas no site de John MacFarlane (MacFarlane, 2015).

C.3 Gerhard Gentzen

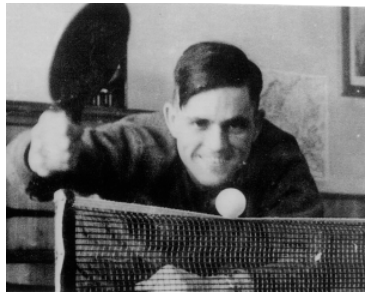


Fig. C.3: Gerhard Gentzen

Gerhard Gentzen é conhecido principalmente como o criador da teoria estrutural da prova e, especificamente, pela criação dos sistemas de derivação de dedução natural e de cálculo de sequentes. Ele nasceu em 24 de novembro de 1909 em Greifswald, na Alemanha. Gerhard foi educado em casa por três anos antes de frequentar o ensino preparatório, onde estava atrás da maioria de seus colegas em termos de instrução. Apesar disso, era um aluno brilhante e demonstrava forte aptidão para a matemática. Seus interesses eram variados e ele, por exemplo, também escrevia poemas para a mãe e peças para o teatro da escola.

Gentzen começou seus estudos universitários na Universidade de Greifswald, mas passou por Göttingen, Munique e Berlim. Recebeu seu doutorado em 1933 pela Universidade de Göttingen, sob a orientação de Hermann Weyl. (Paul Bernays supervisionou a maior parte de seu trabalho, mas foi demitido da universidade pelos nazistas.) Em 1934, Gentzen começou a trabalhar como assistente de David Hilbert. Nesse mesmo ano, desenvolveu os sistemas de derivação de cálculo de sequentes e de dedução natural, em seus artigos *Untersuchungen über das logische Schließen I–II [Investigações sobre a Dedução Lógica I–II].* Ele provou a consistência dos axiomas de Peano em 1936.

A relação de Gentzen com os nazistas é complicada. Ao mesmo tempo em que seu mentor Bernays foi forçado a deixar

a Alemanha, Gentzen ingressou no braço universitário da SA, a organização paramilitar nazista. Como muitos alemães, foi membro do partido nazista. Durante a guerra, serviu como oficial de telecomunicações na unidade de inteligência aérea. No entanto, em 1942 foi dispensado do serviço devido a um colapso nervoso. Não está claro se as lealdades de Gentzen estavam com o partido nazista ou se ele ingressou no partido a fim de garantir sucesso acadêmico.

Em 1943, foi oferecida a Gentzen uma posição acadêmica no Instituto Matemático da Universidade Alemã de Praga, que ele aceitou. No entanto, em 1945 os cidadãos de Praga se revoltaram contra a ocupação alemã. As forças soviéticas chegaram à cidade e prenderam todos os professores da universidade. Por causa de sua filiação a organizações nazistas, Gentzen foi levado a um campo de trabalhos forçados. Ele morreu de desnutrição em sua cela em 4 de agosto de 1945, aos 35 anos.

Leitura complementar Para uma biografia completa de Gentzen, veja Menzler-Trott (2007). Uma leitura interessante sobre matemáticos sob o regime nazista, que traz uma breve nota sobre a vida de Gentzen, é dada por Segal (2014). Os artigos de Gentzen sobre dedução lógica estão disponíveis no alemão original (Gentzen, 1935a,b). Traduções para o inglês dos artigos de Gentzen foram reunidas em um único volume por Szabo (1969), que também inclui um esboço biográfico.

C.4 Kurt Gödel

Kurt Gödel (GER-dle) nasceu em 28 de abril de 1906 em Brünn, no império austro-húngaro (hoje Brno, na República Tcheca). Devido a seu espírito inquisitivo e brilhante, o jovem Kurtele era muitas vezes chamado "Der kleine Herr Warum" (o Pequeno Senhor Porquê) pela família. Destacou-se nos estudos desde a escola primária, tendo obtido menos do que a nota mais alta apenas em matemática. Gödel faltava frequentemente à escola

por causa da frágil saúde e estava isento de educação física. Foi diagnosticado com febre reumática durante a infância. Ao longo da vida, acreditou que isso tinha afetado permanentemente seu coração, apesar dos pareceres médicos em contrário.

Fig. C.4: Kurt Gödel

Gödel começou a estudar na Universidade de Viena em 1924 e concluiu o doutorado em 1929. Inicialmente pretendia estudar física, mas seus interesses se deslocaram rapidamente para a matemática e sobretudo para a lógica, em parte devido à influência do filósofo Rudolf Carnap. Sua dissertação, orientada por Hans Hahn, provava o teorema de completude da lógica de predicados de primeira ordem com identidade (Gödel, 1929). Apenas um ano depois, obteve seus resultados mais famosos—o primeiro e o segundo teorema da incompletude (publicados em Gödel 1931). Durante o período em Viena, Gödel participou intensamente no Círculo de Viena, um grupo de filósofos de orientação científica que incluía Carnap, cujo trabalho foi especialmente influenciado pelos resultados de Gödel.

Em 1938, Gödel se casou com Adele Nimbursky. Os pais não ficaram contentes: não só ela era seis anos mais velha e já divorciada, como trabalhava como bailarina num clube noturno. As pressões sociais não afetaram Gödel, contudo, e permaneceram felizmente casados até sua morte.

Depois de a Alemanha Nazi ter anexado a Áustria em 1938, Gödel e Adele emigraram para os Estados Unidos, onde ele assumiu um lugar no Institute for Advanced Study em Princeton,

Nova Jersey. Apesar da introversão e do caráter excêntrico, o período de Gödel em Princeton foi colaborativo e frutuoso. Publicou ensaios em teoria dos conjuntos, filosofia e física. Notavelmente, estabeleceu uma amizade particularmente forte com o colega no IAS, Albert Einstein.

Nos últimos anos, a saúde mental de Gödel se deteriorou. A internação da esposa em 1977 fez com que ela não pudesse mais cozinhar as refeições dele. Tendo sofrido de problemas de saúde mental ao longo da vida, sucumbiu à paranoia. Com medo mortal de ser envenenado, Gödel se recusou a comer. Morreu de inanição em 14 de janeiro de 1978, em Princeton.

Leitura complementar Para uma biografia completa da vida de Gödel, ver John Dawson (1997). Para outros textos biográficos, bem como ensaios sobre as contribuições de Gödel para a lógica e a filosofia, ver Wang (1990), Baaz et al. (2011), Takeuti et al. (2003), e Sigmund et al. (2007).

A tese de doutorado de Gödel está disponível no alemão original (Gödel, 1929). O texto original dos teoremas da incompletude é (Gödel, 1931). Todas as publicações e textos não publicados de Gödel, bem como uma seleção de correspondência, estão disponíveis em inglês em seus *Collected Papers* Feferman et al. (1986, 1990).

Para um tratamento detalhado dos teoremas da incompletude de Gödel, ver Smith (2013). Para uma discussão informal e filosófica dos teoremas de Gödel, ver o podcast de Mark Linsenmayer (Linsenmayer, 2014).

C.5 Emmy Noether

Emmy Noether (NER-ter) nasceu em Erlangen, na Alemanha, em 23 de março de 1882, em uma família erudita de classe média alta. Aclamada como a "mãe da álgebra moderna", Noether fez contribuições inovadoras tanto para a matemática quanto para a física, apesar das significativas barreiras à educação das mu-

lheres. Na Alemanha daquela época, esperava-se que as meninas fossem educadas nas artes e elas não tinham permissão para frequentar escolas preparatórias para a faculdade. No entanto, após assistir como ouvinte às aulas nas Universidades de Göttingen e Erlangen (onde seu pai era professor de matemática), Noether finalmente conseguiu se matricular como aluna em Erlangen em 1904, quando a política da instituição foi atualizada para permitir estudantes do sexo feminino. Ela recebeu seu doutorado em matemática em 1907.

Fig. C.5: Emmy Noether

Apesar de suas qualificações, Noether enfrentou muita resistência durante sua carreira. De 1908 a 1915, ela lecionou em Erlangen sem remuneração. Durante esse período, chamou a atenção de David Hilbert, um dos mais eminentes matemáticos do mundo na época, que a convidou para Göttingen. No entanto, as mulheres eram proibidas de obter cátedras, e ela só pôde lecionar sob o nome de Hilbert, novamente sem remuneração. Foi nesse período que ela provou o que hoje é conhecido como o teorema de Noether, que ainda é usado na física teórica atualmente. Noether finalmente recebeu o direito de lecionar em 1919. A resposta de Hilbert à resistência contínua de seus colegas de universidade teria sido: "Senhores, o conselho da faculdade não é uma casa de banhos."

No final da década de 1920, ela concentrou-se no trabalho em álgebra abstrata, e suas contribuições revolucionaram o campo. Em suas provas, ela frequentemente fazia uso da chamada condição de cadeia ascendente, que afirma que não há cadeia infinita

estritamente crescente de certos conjuntos. Por exemplo, certas estruturas algébricas hoje conhecidas como anéis noetherianos têm a propriedade de que não há sequências infinitas de ideais $I_1 \subsetneq I_2 \subsetneq \ldots$. A condição pode ser generalizada para qualquer ordem parcial (na álgebra, ela diz respeito ao caso especial de ideais ordenados pela relação de subconjunto), e também podemos considerar a condição dual de cadeia descendente, na qual toda sequência estritamente *de*crescente em uma ordem parcial eventualmente termina. Se uma ordem parcial satisfaz a condição de cadeia descendente, é possível usar indução ao longo dessa ordem de maneira semelhante àquela em que podemos usar indução ao longo da ordem $<$ sobre $\mathbb{N}$. Tais ordens são chamadas *bem fundadas* ou *noetherianas*, e o princípio de prova correspondente, *indução noetheriana.*

Noether era judia e, quando os nazistas chegaram ao poder em 1933, ela foi demitida de seu cargo. Felizmente, Noether conseguiu emigrar para os Estados Unidos para ocupar uma posição temporária em Bryn Mawr, na Pensilvânia. Durante seu tempo lá, ela também lecionou em Princeton, embora considerasse a universidade pouco acolhedora para as mulheres (Dick, 1981, 81). Em 1935, Noether passou por uma operação para remover um tumor uterino. Ela morreu de uma infecção decorrente da cirurgia e foi enterrada em Bryn Mawr.

Leitura complementar Para uma biografia de Noether, veja Dick (1981). O Perimeter Institute for Theoretical Physics disponibiliza on-line suas palestras sobre a vida e a influência de Noether (Institute, 2015). Se você estiver cansado de ler, *Stuff You Missed in History Class* tem um podcast sobre a vida e a influência de Noether (Frey e Wilson, 2015). As obras reunidas de Noether estão disponíveis no alemão original (Jacobson, 1983).

C.6 Bertrand Russell

Bertrand Russell é celebrado como um dos fundadores da filosofia analítica moderna. Nascido em 18 de maio de 1872, Russell não era conhecido apenas pelo trabalho em filosofia e lógica, mas escreveu muitos livros de divulgação em diversas áreas. Foi também um ardente ativista político ao longo da vida.

Russell nasceu em Trellech, Monmouthshire, no País de Gales. Os pais pertenciam à nobreza britânica. Eram livre-pensadores e chegaram a travar amizade com os radicais de Boston da época. Infelizmente, os pais de Russell morreram quando ele era ainda criança, e Russell foi viver com os avós. Ali recebeu uma educação religiosa (algo que os pais tinham querido evitar a todo custo). A avó era muito rigorosa em todas as questões de moralidade. Na adolescência foi sobretudo educado em casa por professores particulares.

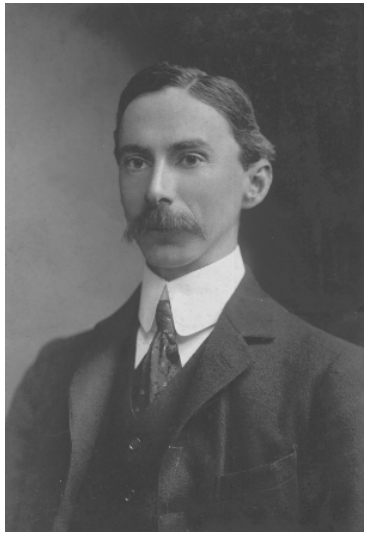


Fig. C.6: Bertrand Russell

A influência de Russell na filosofia analítica, e sobretudo na lógica, é enorme. Estudou matemática e filosofia no Trinity College, em Cambridge, onde foi influenciado pelo matemático e filósofo Alfred North Whitehead. Em 1910, Russell e Whitehead publicaram o primeiro volume de *Principia Mathematica*, defendendo a ideia de que a matemática é redutível à lógica. Seguiu-se a publicação de centenas de livros, ensaios e panfletos políticos. Em 1950, ganhou o Prêmio Nobel de Literatura.

Russell se encontrava profundamente envolvido na política e no ativismo social. Durante a Primeira Guerra Mundial foi preso e condenado a seis meses de prisão por atividades pacifistas e protesto. Na prisão pôde escrever e ler, e afirma ter achado a

experiência "bastante agradável." Permaneceu pacifista ao longo da vida, e voltou a ser detido por participar num comício pelo desarmamento nuclear em 1961. Sobreviveu também a um acidente de avião em 1948, no qual os únicos sobreviventes eram os que estavam sentados na área de fumantes. Por isso, Russell declarou que devia a vida ao tabagismo. Russell se casou quatro vezes, mas tinha fama de manter relações extraconjugais. Morreu em 2 de fevereiro de 1970, aos 97 anos, em Penrhyndeudraeth, no País de Gales.

Leitura complementar Russell escreveu uma autobiografia em três partes, que abrange sua vida de 1872 a 1967 (Russell, 1967, 1968, 1969). O Bertrand Russell Research Centre na McMaster University alberga os arquivos de Bertrand Russell. Ver o site em Duncan (2015), para informação sobre os volumes de suas obras reunidas (incluindo índices pesquisáveis) e projetos de arquivo. O artigo de Russell *On Denoting* (Russell, 1905) é um clássico da filosofia analítica do século XX.

A entrada da Stanford Encyclopedia of Philosophy sobre Russell (Irvine, 2015) inclui excertos de áudio de Russell a falar sobre Desejo e teoria política. Estão disponíveis online muitas entrevistas em vídeo com Russell. Para o ouvir falar de tabagismo e do envolvimento num acidente de avião, ver, por exemplo, Russell (n.d.). Algumas obras de Russell, incluindo a *Introduction to Mathematical Philosophy*, estão disponíveis como audiolivros gratuitos em LibriVox (n.d.).

C.7 Alfred Tarski

Alfred Tarski nasceu em 14 de janeiro de 1901 em Varsóvia, na Polônia (então parte do Império Russo). Descrito como "napoleônico", Tarski era ruidoso, falador e intenso. Sua energia se refletia muitas vezes nas aulas—chegou a incendiar um cesto de lixo ao jogar fora um cigarro durante uma aula, e foi proibido de lecionar novamente naquele edifício.

Fig. C.7: Alfred Tarski

Tarski tinha sede de conhecimento desde cedo. Mais tarde diria aos estudantes que estudara lógica porque era a única disciplina em que tivera um B, mas os registros do colégio mostram que tinha notas máximas em tudo—até em lógica. Estudou na Universidade de Varsóvia de 1918 a 1924. Tarski pretendia inicialmente estudar biologia, mas se interessou por matemática, filosofia e lógica, pois a universidade era o centro da Escola de Lógica e Filosofia de Varsóvia. Tarski se doutorou em 1924 sob orientação de Stanisław Leśniewski.

Antes de emigrar para os Estados Unidos em 1939, Tarski completou parte de seu trabalho mais importante enquanto professor de ensino secundário em Varsóvia. O trabalho sobre consequência lógica e verdade lógica foi escrito nesse período. Em 1939, Tarski se encontrava nos Estados Unidos numa turnê de palestras. Durante a visita, a Alemanha invadiu a Polônia, e, por causa da ascendência judaica, Tarski não pôde retornar. A esposa e os filhos permaneceram na Polônia até o fim da guerra, mas depois também puderam emigrar para os Estados Unidos. Tarski lecionou em Harvard, no College of the City of New York, no Institute for Advanced Study em Princeton, e finalmente na Universidade da Califórnia, em Berkeley. Ali fundou o programa multidisciplinar em Lógica e Metodologia da Ciência. Tarski morreu em 26 de outubro de 1983, aos 82 anos.

Leitura complementar Para mais sobre a vida de Tarski, ver a biografia *Alfred Tarski: Life and Logic* (Feferman e Feferman,

2004). As obras fundadoras de Tarski sobre consequência lógica e verdade estão disponíveis em inglês em (Corcoran, 1983). Todas as obras originais de Tarski foram reunidas numa série de quatro volumes, (Tarski, 1981).

C.8 Alan Turing

Alan Turing nasceu em Maida Vale, Londres, em 23 de junho de 1912. É considerado o pai da ciência da computação teórica. O interesse de Turing pelas ciências físicas e pela matemática começou em tenra idade. No entanto, quando menino, seus interesses não eram bem contemplados nas escolas que frequentava, onde a ênfase recaía sobre a literatura e os clássicos. Consequentemente, ele teve um desempenho fraco na escola e foi repreendido por muitos de seus professores.

Turing estudou no King's College, em Cambridge, na graduação, onde estudou matemática. Em 1936, Turing desenvolveu (o que hoje se chama) a máquina de Turing como uma tentativa de definir precisamente a noção de função computável e de provar a indecidibilidade do problema da decisão. Ele foi superado no resultado por Alonzo Church, que o provou por meio de seu próprio cálculo lambda. Ainda assim, o artigo de Turing foi publicado com referência ao resultado de Church. Church convidou Turing para Princeton, onde ele passou de 1936 a 1938 e obteve um doutorado sob a orientação de Church.

Fig. C.8: Alan Turing

Apesar de seu interesse pela lógica, os interesses anteriores de Turing pelas ciências físicas permaneceram predominantes. Suas habilidades práticas foram postas em prática durante seu serviço no departamento britânico de criptanálise em Bletchley Park durante a Segunda Guerra Mundial. Turing foi uma figura central na quebra da cifra usada nas comunicações navais alemãs—o código Enigma. A expertise de Turing em estatística e criptografia, juntamente com a introdução de maquinário eletrônico, deu à equipe a capacidade de quebrar o código por meio da criação de uma máquina de descriptografia chamada "bombe". Suas ideias também ajudaram na criação do primeiro computador eletrônico programável do mundo, o Colossus, também usado em Bletchley Park para quebrar a cifra alemã Lorenz.

Turing era homossexual. Mesmo assim, em 1942 ele pediu Joan Clarke em casamento, uma de suas colegas de equipe em Bletchley Park, mas depois rompeu o noivado e confessou a ela que era homossexual. Teve vários amantes ao longo da vida, embora os atos homossexuais fossem, então, crimes no Reino Unido. Em 1952, a casa de Turing foi assaltada por um amigo de seu amante da época e, ao prestar queixa à polícia, Turing admitiu ter um relacionamento homossexual, sob a impressão de que o governo estava a caminho de legalizar os atos homossexuais. Isso não era verdade, e ele foi acusado de grave indecência. Em vez de ir para a prisão, Turing optou por um tratamento hormonal que reduzia a libido. Turing foi encontrado morto em 8 de junho de 1954, de uma overdose de cianeto—muito provavelmente suicídio. Ele recebeu um perdão real da Rainha Elizabeth II em 2013.

Leitura complementar Para uma biografia abrangente de Alan Turing, veja Hodges (2014). A vida e a obra de Turing inspiraram uma peça, *Breaking the Code*, que foi produzida em 1996 para a TV, com Derek Jacobi no papel de Turing. *The Imitation Game*, um filme indicado ao Oscar, com Bendict Cumberbatch e

Kiera Knightley, também é baseado livremente na vida de Alan Turing e em seu período em Bletchley Park (Tyldum, 2014).

Radiolab (2012) tem vários podcasts sobre a vida e a obra de Turing. O documentário *The Strange Life and Death of Dr. Turing*, da BBC Horizon, está disponível para assistir on-line (Sykes, 1992). (Theelen, 2012) é um vídeo curto de uma Máquina de Turing de LEGO em funcionamento—feito para homenagear o centenário de Turing em 2012.

O artigo original de Turing sobre máquinas de Turing e o problema da decisão é Turing (1937).

C.9 Ernst Zermelo

Ernst Zermelo nasceu em 27 de julho de 1871 em Berlim, na Alemanha. Tinha cinco irmãs, embora a família sofresse de frágil saúde e apenas três sobrevivessem à idade adulta. Os pais também faleceram quando ele era jovem, deixando-o órfão, com os irmãos, aos dezessete anos. Zermelo tinha um profundo interesse pelas artes, e sobretudo pela poesia. Era conhecido por ser perspicaz, espirituoso e crítico. Entre suas conquistas matemáticas mais celebradas estão a introdução do axioma da escolha (em 1904) e sua axiomatização da teoria dos conjuntos (em 1908).

Fig. C.9: Ernst Zermelo

Os interesses de Zermelo na universidade eram variados. Frequentou cursos de física, matemática e filosofia. Sob orientação de Hermann Schwarz, Zermelo concluiu a dissertação *Investiga-*

tions in the Calculus of Variations em 1894 na Universidade de Berlim. Em 1897, decidiu prosseguir estudos na Universidade de Göttingen, onde foi fortemente influenciado pelo trabalho fundacional de David Hilbert. Em 1899 tornou-se elegível para uma cátedra, mas só obteve uma onze anos depois—possivelmente devido a seu comportamento estranho e à "pressa nervosa."

Zermelo obteve finalmente uma cátedra remunerada na Universidade de Zurique em 1910, mas foi forçado a se aposentar em 1916 por causa da tuberculose. Depois da recuperação, foi nomeado professor honorário na Universidade de Freiburg em 1921. Nesse período dedicou-se à matemática fundacional. Irritou-se com os trabalhos de Thoralf Skolem e Kurt Gödel, e criticou publicamente suas abordagens em seus artigos. Foi demitido do cargo em Freiburg em 1935, devido a sua impopularidade e à oposição à ascensão de Hitler ao poder na Alemanha.

Os últimos anos da vida de Zermelo foram marcados pelo isolamento. Depois da demissão em 1935, abandonou a matemática. Mudou-se para o campo, onde viveu modestamente. Casou-se em 1944 e ficou completamente dependente da esposa à medida que ia ficando cego. Zermelo perdeu totalmente a visão por volta de 1951. Faleceu em Günterstal, na Alemanha, em 21 de maio de 1953.

Leitura complementar Para uma biografia completa de Zermelo, ver Ebbinghaus (2015). Os artigos fundadores de 1904 e 1908 de Zermelo podem ser lidos no alemão original (Zermelo, 1904, 1908). As obras reunidas de Zermelo, incluindo os escritos sobre física, estão disponíveis em tradução inglesa em Ebbinghaus et al. (2010); Ebbinghaus e Kanamori (2013).

APÊNDICE D

O Alfabeto Grego

Alfa	α	A	Niu	ν	N
Beta	β	B	Xi	ξ	Ξ
Gama	γ	Γ	Ômicron	o	O
Delta	δ	Δ	Pi	π	Π
Épsilon	ε	E	Rô	ρ	P
Zeta	ζ	Z	Sigma	σ	Σ
Eta	η	H	Tau	τ	T
Teta	θ	Θ	Ípsilon	υ	Υ
Iota	ι	I	Fi	φ	Φ
Capa	κ	K	Qui	χ	X
Lambda	λ	Λ	Psi	ψ	Ψ
Mi	μ	M	Ômega	ω	Ω

Glossário

anti-simétrica R é anti-simétrica se e somente se, sempre que valem ambos Rxy e Ryx, então $x = y$; em outras palavras: se $x \neq y$, então não Rxy ou não Ryx (veja seção 2.2).

assimétrica R é assimétrica se, para nenhum par $x, y \in A$, temos Rxy e Ryx (veja seção 2.4).

atribuição de variáveis Uma função que mapeia cada variável a um elemento de $|M|$ (veja seção 7.4).

bijeção Uma função que é tanto sobrejetora quanto injetora (veja seção 3.2).

coberta Uma estrutura na qual todo elemento do domínio é o valor de algum termo fechado (veja seção 7.2).

completude Propriedade de um sistema de derivação; é completo se, sempre que A é consequência de Γ, então há também uma derivação que estabelece $\Gamma \vdash A$; equivalentemente, se e somente se todo conjunto consistente de sentenças é satisfatível (veja seção 12.1).

composição $(g \circ f)$ A função resultante de "encadear" f e g; $(g \circ f)(x) = g(f(x))$ (veja seção 3.5).

conexa R é conexa se, para todos $x, y \in A$ com $x \neq y$, então Rxy ou Ryx (veja seção 2.2).

conjunto Uma coleção de objetos, considerada independentemente da maneira como é especificada, da ordem dos objetos no conjunto, e de sua multiplicidade (veja seção 1.1).

conjunto consistente completo Um conjunto de sentenças é completo e consistente se e somente se é consistente e, para toda sentença A, ou A ou $\neg A$ está no conjunto (veja seção 12.3).

conjunto potência ($\wp(A)$) O conjunto que consiste em todos os subconjuntos de um conjunto A, $\wp(A) = \{x : x \subseteq A\}$ (veja seção 1.2).

consequência ($\Gamma \models A$) Uma sentença A é consequência de um conjunto de sentenças Γ se e somente se, para toda estrutura $\mathfrak{M}$ com $\mathfrak{M} \models \Gamma$, $\mathfrak{M} \models A$ (veja seção 7.7).

consistente No cálculo de sequentes, um conjunto de sentenças Γ é consistente se e somente se não há derivação de um sequente $\Gamma_0 \Rightarrow$ com $\Gamma_0 \subseteq \Gamma$ (veja seção 10.8). Na dedução natural, Γ é consistente se e somente se $\Gamma \nvdash \bot$ (veja seção 11.7). Se Γ não é consistente, é inconsistente..

correção Propriedade de um sistema de derivação: é correto se, sempre que $\Gamma \vdash A$, então $\Gamma \models A$ (veja seção 10.12 e seção 11.11).

derivabilidade ($\Gamma \vdash A$) No cálculo de sequentes, A é derivável de Γ se há uma derivação de um sequente $\Gamma_0 \Rightarrow A$ em que $\Gamma_0 \subseteq \Gamma$ é uma sequência finita de sentenças em Γ (veja seção 10.8). Na dedução natural, A é derivável de Γ se há uma derivação com fórmula final A e na qual toda hipótese é descarregada ou está em Γ (veja seção 11.7).

derivação No cálculo de sequentes, uma árvore de sequentes na qual todo sequente é um sequente inicial ou segue dos sequentes imediatamente acima dele por uma regra de inferência (veja seção 10.1). Na dedução natural, uma árvore de fórmulas na qual toda fórmula é uma hipótese ou segue das fórmulas imediatamente acima dela por uma regra de inferência (veja seção 11.1).

descarregada Uma hipótese em uma derivação pode ser descarregada por uma regra de inferência abaixo dela (a regra e a hipótese recebem então um rótulo correspondente,

por exemplo, $[A]^2$). Se não é descarregada, é chamada de não descarregada (veja seção 11.1).

diferença ($A \setminus B$) o conjunto de todos os elementos de A que não são também elementos de B: $A \setminus B = \{x : x \in A \text{ e } x \notin B\}$ (veja seção 1.4).

disjuntos dois conjuntos sem elementos em comum (veja seção 1.4).

domínio (de uma estrutura) ($|M|$) Conjunto não vazio do qual uma estrutura toma atribuições e valores de variáveis (veja seção 7.2).

domínio (de uma função) ($\mathrm{dom}(f)$) O conjunto de objetos para os quais uma função (parcial) está definida (veja seção 3.1).

enumeração Uma lista possivelmente infinita de todos os elementos de um conjunto A; formalmente, uma função sobrejetora $f : \mathbb{N} \to A$ (veja seção 4.2).

equinumeroso A e B são equinumerosos se e somente se há uma bijeção total de A para B (veja seção 4.8).

estrutura (M) Uma interpretação de uma linguagem de primeira ordem, consistindo em um domínio (de uma estrutura) e atribuições dos símbolos de constante, de predicado e de função da linguagem (veja seção 7.2).

extensionalidade (da satisfação) Se uma fórmula A é ou não satisfeita depende apenas das atribuições aos símbolos não lógicos e às variáveis livres que de fato ocorrem em A.

extensionalidade (de conjuntos) Os conjuntos A e B são idênticos, $A = B$, se e somente se todo elemento de A também é um elemento de B, e vice-versa (veja seção 1.1).

fechado Um conjunto de sentenças Γ é fechado se e somente se, sempre que $\Gamma \vDash A$, então $A \in \Gamma$. O conjunto $\{A : \Gamma \vDash A\}$ é o fecho de Γ (veja seção 8.1).

fecho transitivo (R^+) a menor relação transitiva que contém R (veja seção 2.7).

finitamente satisfatível Γ é finitamente satisfatível se e somente se todo $\Gamma_0 \subseteq \Gamma$ finito é satisfatível (veja seção 12.9).

função ($f : A \to B$) Um mapeamento de cada elemento de um domínio (de uma função) A a um elemento do contradomínio B (veja seção 3.1).

função inversa Se $f : A \to B$ é uma bijeção, $f^{-1} : B \to A$ é a função com $f^{-}1(y) =$ qualquer que seja o único $x \in A$ tal que $f(x) = y$ (veja seção 3.4).

função parcial ($f : A \nrightarrow B$) Uma função parcial é um mapeamento que atribui a todo elemento de A no máximo um elemento de B. Se f atribui um elemento de B a $x \in A$, $f(x)$ está definida; caso contrário, indefinida (veja seção 3.6).

fórmula Expressões de uma linguagem de primeira ordem $\mathcal{L}$ que expressam relações ou propriedades, ou são verdadeiras ou falsas (veja seção 6.3).

grafo (de uma função) a relação $R_f \subseteq A \times B$ definida por $R_f = \{\langle x, y\rangle : f(x) = y\}$, se $f : A \nrightarrow B$ (veja seção 3.3).

hipótese Uma fórmula que está no topo de uma derivação, também chamada de fórmula inicial. Pode ser descarregada ou não descarregada (veja seção 11.1).

imagem ($\mathrm{Im}(f)$) o subconjunto do contradomínio que é de fato gerado como saída por f; $\mathrm{Im}(f) = \{y \in B : f(x) = y$ para algum $x \in A\}$ (veja seção 3.1).

inconsistente veja consistente.

injetora $f : A \to B$ é injetora se e somente se, para cada $y \in B$, há no máximo um $x \in A$ tal que $f(x) = y$; equivalentemente, se, sempre que $x \neq x'$, então $f(x) \neq f(x')$ (veja seção 3.2).

interseção ($A \cap B$) O conjunto de todas as coisas que são elementos tanto de A quanto de B: $A \cap B = \{x : x \in A \wedge x \in B\}$ (veja seção 1.4).

irreflexiva R é irreflexiva se, para nenhum $x \in A$, Rxx (veja seção 2.4).

ligada Ocorrência de uma variável dentro do escopo de um quantificador que usa a mesma variável (veja seção 6.8).

livre Uma ocorrência de uma variável que não é ligada (veja seção 6.8).

livre para Um termo t é livre para x em A se nenhuma das ocorrências livre de x em A ocorre no escopo de um quantificador que liga uma variável em t (veja seção 6.9).

modelo Uma estrutura na qual toda sentença em Γ é verdadeira é um modelo de Γ (veja seção 8.2).

não descarregada veja descarregada.

ordem estrita Uma relação irreflexiva, assimétrica e transitiva (veja seção 2.4).

ordem linear Uma ordem parcial conexa (veja seção 2.4).

ordem linear estrita Uma ordem estrita conexa (veja seção 2.4).

ordem parcial Uma relação reflexiva, anti-simétrica e transitiva (veja seção 2.4).

ordem total veja ordem linear.

problema da decisão Problema de decidir se uma dada sentença da lógica de primeira ordem é válida ou não (veja Teorema de Church-Turing).

problema da parada O problema de determinar (para quaisquer e, n) se a máquina de Turing M_e para ao receber uma entrada de n traços (veja seção 15.4).

produto cartesiano $(A \times B)$ Conjunto de todos os pares de elementos de A e B; $A \times B = \{\langle x, y\rangle : x \in A \text{ e } y \in B\}$ (veja seção 1.5).

pré-ordem Uma relação reflexiva e transitiva (veja seção 2.4).

reflexiva R é reflexiva se e somente se, para todo $x \in A$, Rxx (veja seção 2.2).

relação binária Um subconjunto de A^2; escrevemos Rxy (ou xRy) para $\langle x, y\rangle \in R$ (veja seção 2.1).

relação de equivalência uma relação reflexiva, simétrica e transitiva (veja seção 2.2).

relação inversa (R^{-1}) A relação R "invertida"; $R^{-1} = \{\langle y, x\rangle : \langle x, y\rangle \in R\}$ (veja seção 2.7).

satisfatível Um conjunto de sentenças Γ é satisfatível se $\boldsymbol{M} \vDash \Gamma$ para alguma estrutura $\boldsymbol{M}$; caso contrário, é insatisfatível (veja seção 7.7).

sentença Uma fórmula sem nenhuma variável livre. (veja seção 6.8).

sequente Uma expressão da forma $\Gamma \Rightarrow \Delta$ em que Γ e Δ são sequências finitas de sentenças (veja seção 10.1).

sequência (finita) (A^*) Uma cadeia finita de elementos de A; um elemento de A^n para algum n (veja seção 1.3).

sequência (infinita) (A^ω) Uma sequência sem lacunas e interminável de elementos de A; formalmente, uma função $s\colon \mathbb{Z}^+ \to A$ (veja seção 1.3).

simétrica R é simétrica se e somente se, sempre que Rxy, então também Ryx (veja seção 2.2).

sobrejetora $f\colon A \to B$ é sobrejetora se e somente se a imagem de f é todo o B, isto é, para todo $y \in B$ há pelo menos um $x \in A$ tal que $f(x) = y$ (veja seção 3.2).

subconjunto ($A \subseteq B$) Um conjunto do qual todo elemento é um elemento de um dado conjunto B (veja seção 1.2).

subfórmula Parte de uma fórmula que é, ela mesma, uma fórmula (veja seção 6.6).

teorema ($\vdash A$) No cálculo de sequentes, uma fórmula A é um teorema (da lógica) se há uma derivação do sequente $\Rightarrow A$ (veja seção 10.8). Na dedução natural, uma fórmula A é um teorema se há uma derivação de A com toda hipótese descarregada (veja seção 11.7). Também dizemos que A é um teorema de uma teoria Γ se $\Gamma \vdash A$.

teorema da compacidade Afirma que todo conjunto finitamente satisfatível de sentenças é satisfatível (veja seção 12.9).

teorema da completude Afirma que a lógica de primeira ordem é completa: todo conjunto consistente de sentenças é satisfatível.

teorema da dedução Relaciona a consequência e a derivabilidade de uma sentença a partir de uma hipótese com os de um condicional correspondente. Na forma semântica (Teorema 7.30), afirma que $\Gamma \cup \{A\} \vDash B$ se e somente se $\Gamma \vDash A \to B$. Na forma teórico-demonstrativa, afirma que $\Gamma \cup \{A\} \vdash B$ se e somente se $\Gamma \vdash A \to B$.

Teorema de Church-Turing Afirma que não há máquina de Turing que decida se uma dada sentença da lógica de primeira ordem é válida ou não (veja seção 15.8).

Teorema de Löwenheim-Skolem Afirma que todo conjunto satisfatível de sentenças tem um modelo contável (veja seção 12.11).

Tese de Church-Turing afirma que tudo o que é computável por meio de um procedimento efetivo é Turing-computável (veja seção 14.10).

transitiva R é transitiva se e somente se, sempre que Rxy e Ryz, então também Rxz (veja seção 2.2).

união ($A \cup B$) O conjunto de todos os elementos de A e B juntos: $A \cup B = \{x : x \in A \vee x \in B\}$ (veja seção 1.4).

variável própria No cálculo de sequentes, um símbolo de constante especial em uma premissa de uma inferência $\exists$L ou $\forall$R que não pode aparecer na conclusão (veja seção 10.1). Na dedução natural, um símbolo de constante especial em uma premissa de uma inferência $\exists$Elim ou $\forall$Intro que não pode aparecer na conclusão nem em qualquer hipótese não descarregada (veja seção 11.1).

válida ($\vDash A$) Uma sentença A é *válida* se e somente se $\mathfrak{M} \vDash A$ para toda estrutura $\mathfrak{M}$ (veja seção 7.7).

x-variante Duas atribuições de variáveis são x-variantes, $s \sim_x s'$, se diferem no máximo no que atribuem a x (veja seção 7.4).

Créditos das Imagens

Georg Cantor, p. 402: Retrato de Georg Cantor por Otto Zeth, cortesia do Arquivo Universitário, Martin-Luther-Universität Halle–Wittenberg. UAHW Rep. 40-VI, Nr. 3 Bild 102.

Kurt Gödel, p. 407: Retrato de Kurt Gödel, c. 1925, fotógrafo desconhecido. Do Shelby White and Leon Levy Archives Center, Institute for Advanced Study, Princeton, NJ, EUA, em depósito na Biblioteca da Universidade de Princeton, Divisão de Manuscritos, Departamento de Livros Raros e Coleções Especiais, Kurt Gödel Papers, (C0282), Box 14b, #110000. O *Open Logic Project* obteve permissão do Arquivo do Instituto para usar esta imagem em materiais derivados do OLP não comerciais. É necessária autorização do Arquivo para qualquer outro uso.

Bertrand Russell, p. 411: Retrato de Bertrand Russell, c. 1907, cortesia da Divisão William Ready de Arquivos e Coleções de Investigação, Biblioteca da McMaster University. Arquivos Bertrand Russell, Box 2, f. 4.

Alfred Tarski, p. 413: Fotografia de passaporte de Alfred Tarski, 1939. Recortada e restaurada a partir de digitalização do passaporte de Tarski por Joel Fuller. Original cortesia da Biblioteca Bancroft, University of California, Berkeley. Alfred Tarski Papers, Banc MSS 84/49. O *Open Logic Project* obteve permissão

para usar esta imagem em materiais derivados do OLP não comerciais. É necessária autorização da Biblioteca Bancroft para qualquer outro uso.

Ernst Zermelo, p. 416: Retrato de Ernst Zermelo, c. 1922, cortesia da Abteilung für Handschriften und Seltene Drucke, Niedersächsische Staats- und Universitätsbibliothek Göttingen, Cod. Ms. D. Hilbert 754, Bl. 6 Nr. 25.

Referências Bibliográficas

Aspray, William. 1984. The Princeton mathematics community in the 1930s: Alonzo Church. URL http://www.princeton.edu/mudd/finding_aids/mathoral/pmc05.htm. Interview.

Baaz, Matthias, Christos H. Papadimitriou, Hilary W. Putnam, Dana S. Scott, e Charles L. Harper Jr. 2011. *Kurt Gödel and the Foundations of Mathematics: Horizons of Truth*. Cambridge: Cambridge University Press.

Cantor, Georg. 1892. Über eine elementare Frage der Mannigfaltigkeitslehre. *Jahresbericht der deutschen Mathematiker-Vereinigung* 1: 75–8.

Cheng, Eugenia. 2004. How to write proofs: A quick quide. URL https://eugeniacheng.com/wp-content/uploads/2017/02/cheng-proofguide.pdf.

Church, Alonzo. 1936a. A note on the Entscheidungsproblem. *The Journal of Symbolic Logic* 1: 40–41.

Church, Alonzo. 1936b. An unsolvable problem of elementary number theory. *American Journal of Mathematics* 58: 345–363.

Corcoran, John. 1983. *Logic, Semantics, Metamathematics.* Indianapolis: Hackett, 2nd ed.

Dauben, Joseph. 1990. *Georg Cantor: His Mathematics and Philosophy of the Infinite.* Princeton: Princeton University Press.

Dick, Auguste. 1981. *Emmy Noether 1882–1935.* Boston: Birkhäuser.

du Sautoy, Marcus. 2014. A brief history of mathematics: Georg Cantor. URL http://www.bbc.co.uk/programmes/b00ss1j0. Audio Recording.

Duncan, Arlene. 2015. The Bertrand Russell Research Centre. URL http://russell.mcmaster.ca/.

Ebbinghaus, Heinz-Dieter. 2015. *Ernst Zermelo: An Approach to his Life and Work.* Berlin: Springer-Verlag.

Ebbinghaus, Heinz-Dieter, Craig G. Fraser, e Akihiro Kanamori. 2010. *Ernst Zermelo. Collected Works*, vol. 1. Berlin: Springer-Verlag.

Ebbinghaus, Heinz-Dieter e Akihiro Kanamori. 2013. *Ernst Zermelo: Collected Works*, vol. 2. Berlin: Springer-Verlag.

Enderton, Herbert B. 2019. Alonzo Church: Life and Work. Em *The Collected Works of Alonzo Church*, orgs. Tyler Burge e Herbert B. Enderton. Cambridge, MA: MIT Press.

Feferman, Anita e Solomon Feferman. 2004. *Alfred Tarski: Life and Logic.* Cambridge: Cambridge University Press.

Feferman, Solomon, John W. Dawson Jr., Stephen C. Kleene, Gregory H. Moore, Robert M. Solovay, e Jean van Heijenoort. 1986. *Kurt Gödel: Collected Works. Vol. 1: Publications 1929–1936.* Oxford: Oxford University Press.

Feferman, Solomon, John W. Dawson Jr., Stephen C. Kleene, Gregory H. Moore, Robert M. Solovay, e Jean van Heijenoort. 1990. *Kurt Gödel: Collected Works. Vol. 2: Publications 1938–1974*. Oxford: Oxford University Press.

Frege, Gottlob. 1884. *Die Grundlagen der Arithmetik: Eine logisch mathematische Untersuchung über den Begriff der Zahl.* Breslau: Wilhelm Koebner. Translation in Frege (1953).

Frege, Gottlob. 1953. *Foundations of Arithmetic*, org. J. L. Austin. Oxford: Basil Blackwell & Mott, 2nd ed.

Frey, Holly e Tracy V. Wilson. 2015. Stuff you missed in history class: Emmy Noether, mathematics trailblazer. URL https://www.iheart.com/podcast/stuff-you-missed-in-history-cl-21124503/episode/emmy-noether-mathematics-trailblazer-30207491/. Podcast audio.

Gentzen, Gerhard. 1935a. Untersuchungen über das logische Schließen I. *Mathematische Zeitschrift* 39: 176–210. English translation in Szabo (1969), pp. 68–131.

Gentzen, Gerhard. 1935b. Untersuchungen über das logische Schließen II. *Mathematische Zeitschrift* 39: 176–210, 405–431. English translation in Szabo (1969), pp. 68–131.

Gödel, Kurt. 1929. Über die Vollständigkeit des Logikkalküls [On the completeness of the calculus of logic]. Dissertation, Universität Wien. Reprinted and translated in Feferman et al. (1986), pp. 60–101.

Gödel, Kurt. 1931. über formal unentscheidbare Sätze der *Principia Mathematica* und verwandter Systeme I [On formally undecidable propositions of *Principia Mathematica* and related systems I]. *Monatshefte für Mathematik und Physik* 38: 173–198. Reprinted and translated in Feferman et al. (1986), pp. 144–195.

Grattan-Guinness, Ivor. 1971. Towards a biography of Georg Cantor. *Annals of Science* 27(4): 345–391.

Hammack, Richard. 2013. *Book of Proof.* Richmond, VA: Virginia Commonwealth University. URL http://www.people.vcu.edu/~rhammack/BookOfProof/BookOfProof.pdf.

Hodges, Andrew. 2014. *Alan Turing: The Enigma.* London: Vintage.

Hutchings, Michael. 2003. Introduction to mathematical arguments. URL https://math.berkeley.edu/~hutching/teach/proofs.pdf.

Institute, Perimeter. 2015. Emmy Noether: Her life, work, and influence. URL https://www.youtube.com/watch?v=tNNyAyMRsgE. Video Lecture.

Irvine, Andrew David. 2015. Sound clips of Bertrand Russell speaking. URL http://plato.stanford.edu/entries/russell/russell-soundclips.html.

Jacobson, Nathan. 1983. *Emmy Noether: Gesammelte Abhandlungen—Collected Papers.* Berlin: Springer-Verlag.

John Dawson, Jr. 1997. *Logical Dilemmas: The Life and Work of Kurt Gödel.* Boca Raton: CRC Press.

LibriVox. n.d. Bertrand Russell. URL https://librivox.org/author/1508?primary_key=1508&search_category=author&search_page=1&search_form=get_results. Collection of public domain audiobooks.

Linsenmayer, Mark. 2014. The partially examined life: Gödel on math. URL http://www.partiallyexaminedlife.com/2014/06/16/ep95-godel/. Podcast audio.

MacFarlane, John. 2015. Alonzo Church's JSL reviews. URL http://johnmacfarlane.net/church.html.

Magnus, P. D., Tim Button, J. Robert Loftis, Aaron Thomas-Bolduc, Robert Trueman, e Richard Zach. 2021. *Forall x: Calgary. An Introduction to Formal Logic.* Calgary: Open Logic Project, f21 ed. URL https://forallx.openlogicproject.org/.

Menzler-Trott, Eckart. 2007. *Logic's Lost Genius: The Life of Gerhard Gentzen.* Providence: American Mathematical Society.

Potter, Michael. 2004. *Set Theory and its Philosophy.* Oxford: Oxford University Press.

Radiolab. 2012. The Turing problem. URL http://www.radiolab.org/story/193037-turing-problem/. Podcast audio.

Rose, Daniel. 2012. A song about Georg Cantor. URL https://www.youtube.com/watch?v=QUP5Z4Fb5k4. Audio Recording.

Russell, Bertrand. 1905. On denoting. *Mind* 14: 479–493.

Russell, Bertrand. 1967. *The Autobiography of Bertrand Russell*, vol. 1. London: Allen and Unwin.

Russell, Bertrand. 1968. *The Autobiography of Bertrand Russell*, vol. 2. London: Allen and Unwin.

Russell, Bertrand. 1969. *The Autobiography of Bertrand Russell*, vol. 3. London: Allen and Unwin.

Russell, Bertrand. n.d. Bertrand Russell on smoking. URL https://www.youtube.com/watch?v=80oLTiVW_lc. Video Interview.

Sandstrum, Ted. 2019. *Mathematical Reasoning: Writing and Proof.* Allendale, MI: Grand Valley State University. URL https://scholarworks.gvsu.edu/books/7/.

Segal, Sanford L. 2014. *Mathematicians under the Nazis.* Princeton: Princeton University Press.

Sigmund, Karl, John Dawson, Kurt Mühlberger, Hans Magnus Enzensberger, e Juliette Kennedy. 2007. Kurt Gödel: Das Album–The Album. *The Mathematical Intelligencer* 29(3): 73–76.

Smith, Peter. 2013. *An Introduction to Gödel's Theorems*. Cambridge: Cambridge University Press.

Solow, Daniel. 2013. *How to Read and Do Proofs*. Hoboken, NJ: Wiley.

Steinhart, Eric. 2018. *More Precisely: The Math You Need to Do Philosophy*. Peterborough, ON: Broadview, 2nd ed.

Sykes, Christopher. 1992. BBC Horizon: The strange life and death of Dr. Turing. URL https://www.youtube.com/watch?v=gyusnGbBSHE.

Szabo, Manfred E. 1969. *The Collected Papers of Gerhard Gentzen*. Amsterdam: North-Holland.

Takeuti, Gaisi, Nicholas Passell, e Mariko Yasugi. 2003. *Memoirs of a Proof Theorist: Gödel and Other Logicians*. Singapore: World Scientific.

Tarski, Alfred. 1981. *The Collected Works of Alfred Tarski*, vol. I–IV. Basel: Birkhäuser.

Theelen, Andre. 2012. Lego turing machine. URL https://www.youtube.com/watch?v=FTSAiF9AHN4.

Turing, Alan M. 1937. On computable numbers, with an application to the "Entscheidungsproblem". *Proceedings of the London Mathematical Society, 2nd Series* 42: 230–265.

Tyldum, Morten. 2014. The imitation game. Motion picture.

Velleman, Daniel J. 2019. *How to Prove It: A Structured Approach*. Cambridge: Cambridge University Press, 3rd ed.

Wang, Hao. 1990. *Reflections on Kurt Gödel.* Cambridge: MIT Press.

Zermelo, Ernst. 1904. Beweis, daß jede Menge wohlgeordnet werden kann. *Mathematische Annalen* 59: 514–516. English translation in (Ebbinghaus et al., 2010, pp. 115–119).

Zermelo, Ernst. 1908. Untersuchungen über die Grundlagen der Mengenlehre I. *Mathematische Annalen* 65(2): 261–281. English translation in (Ebbinghaus et al., 2010, pp. 189-229).

Sobre o Open Logic Project

O *Open Logic Text* é um livro-texto colaborativo de código aberto de meta-lógica formal e métodos formais, começando em um nível intermediário (ou seja, após um curso introdutório de lógica formal). Embora voltado a um público não matemático (em particular, estudantes de filosofia e ciência da computação), é rigoroso.

A cobertura de alguns tópicos atualmente incluídos pode ainda não estar completa, e muitas seções ainda exigem revisão substancial. Planejamos expandir o texto para cobrir mais tópicos no futuro. Também planejamos adicionar recursos ao texto, como um glossário, uma lista de leituras complementares, notas históricas, imagens, melhores explicações, seções que explicam a relevância dos resultados para filosofia, ciência da computação e matemática, e mais problemas e exemplos. Se você encontrar um erro ou tiver uma sugestão, informe a equipe do projeto.

O projeto opera no espírito do código aberto. O texto não só está disponível gratuitamente; fornecemos o código-fonte LaTeX sob a licença Creative Commons Atribuição, que dá a qualquer pessoa o direito de baixar, usar, modificar, reorganizar, converter e redistribuir nosso trabalho, desde que deem o devido crédito.

Consulte o site do Open Logic Project em openlogicproject.org para informações adicionais.