Sumário

I

Lógica Proposicional

1 Sintaxe

2 Semântica

3 Problema da Satisfatibilidade

Dada: uma proposição φ;

Pergunta-se: φ é satisfatível?

Esse é o enunciado do chamado **Problema da Satisfatibilidade** (ou problema SAT). É um dos mais icônicos e importantes problemas em computação, e é peça-chave em complexidade computacional: foi o primeiro problema provado pertencer à classe NP-Completo por Steven Cook e Leonid Levin, classe que reúne os problemas difíceis de solucionar, mas fáceis de verificar a solução. Nesse capítulo, estudaremos em detalhes 5 métodos algorítmicos que resolvem o problema. Cada seção contém um exemplo de estudo, e é possível conferir mais exemplos de todos os métodos nos Exercícios Resolvidos.

3.1 Método da Tabela-Verdade

Definido pelo filósofo austríaco Ludwig Wittgenstein, o método da Tabela-Verdade consiste em, dada uma proposição, criar uma tabela de valores-verdade, onde as linhas são todas as valorações-verdade possíveis, e as colunas são subexpressões da proposição avaliada. Se, na última coluna (a da própria proposição), o valor 1 aparecer, a proposição é satisfatível.

Preenchemos inicialmente o campo das variáveis, que representa as valorações-verdade, e então usamos esses valores para preencher o resto da tabela.

- $(A \to B) \vee (\neg B \wedge (C \vee A))$ **é tautologia?**

Queremos provar que toda valoração-verdade satisfaz a proposição. Desse modo, ao construir a tabela-verdade, o valor 1 deve aparecer em todas as linhas da última coluna. Temos:

A	B	C	$\neg B$	$C \vee A$	$A \to B$	$\neg B \wedge (C \vee A)$	$(A \to B) \vee (\neg B \wedge (C \vee A))$
0	0	0	1	0	1	0	1
0	0	1	1	1	1	1	1
0	1	0	0	0	1	0	1
0	1	1	0	1	1	0	1
1	0	0	1	1	0	1	1
1	0	1	1	1	0	1	1
1	1	0	0	1	1	0	1
1	1	1	0	1	1	0	1

Todas os valores da última coluna são 1, implicando que todas as valorações-verdade satisfazem a proposição. Desse modo, $(A \to B) \vee (\neg B \wedge (C \vee A))$ é tautologia.

3.1.1 Complexidade computacional

O método da Tabela-Verdade é poderoso no sentido de que dá como resposta quaisquer informações sobre a satisfatibilidade de uma proposição, bastando apenas conferir a última coluna. Porém, para compensar, tamanha expressividade requer um custo bastante caro. Dada uma proposição de n variáveis e k operadores:

- O número de valorações-verdade possíveis da proposição (ou seja, o número de linhas da tabela) é 2^n.

- O número de subexpressões da proposição (ou seja, o número de colunas da tabela) é, no máximo, $2k + 1$ (Lema 1.1).

Desse modo, a complexidade computacional de tempo da tabela é $O(2^n)$. Custos exponenciais são altos demais e indesejáveis, tornando o próprio método da Tabela-Verdade inviável na prática. Porém, dada a solução pronta (por exemplo, uma valoração-verdade que satisfaz uma proposição), podemos conferi-la montando apenas uma linha da tabela, tornando o custo apenas de $O(2k + 1) = O(k)$.

3.1.2 Métodos confiáveis

Apesar de inviável, o método da Tabela-Verdade serviu de base para todos os outros posteriores, devido a sua expressividade. Desse modo, para um método que resolve SAT ser considerado confiável, ele deve atender a dois critérios:

Corretude: a resposta que o método fornece deve ser a mesma que o método da Tabela-Verdade fornece.

Completude: o método não deve rodar indefinidamente, ou seja, deve parar após um número finito de passos e dar uma resposta.

3.2 Método dos Tableaux Analíticos

Em meados de 1950, os lógicos holandês Evert Beth e finlandês Jaakko Hintikka buscaram independentemente uma formulação intuitiva do conceito de valoração-verdade, e desenvolveram a noção filósofica de **mundo possível**. Buscando estabelecer uma ponte entre esses dois conceitos, formularam um método algorítmico que resolve SAT que se baseia em uma **árvore de possibilidades**, a partir de um conjunto de regras simples. Tal método ficou conhecido como método dos Tableaux. Com ausência de contradição, o mundo possível seria revelado por um caminho de uma raiz a uma folha, acarretando em satisfatibilidade. Em 1970, o americano Raymond Smullyan definiu uma metodologia mais eficiente para a aplicação das regras, e então o método ficou conhecido como **método dos Tableaux Analíticos**.

3.2.1 O conjunto de regras

O conjunto de regras é dividido em dois subconjuntos: as regras do tipo α, que simbolizam apenas uma possibilidade, e as regras do tipo β, que simbolizam duas.

	α	β
$\neg$	$\hat{v}(\neg\varphi) = 1$ $\hat{v}(\varphi) = 0$ $\hat{v}(\neg\varphi) = 0$ $\hat{v}(\varphi) = 1$	
$\wedge$	$\hat{v}(\varphi \wedge \psi) = 1$ $\hat{v}(\varphi) = 1$ $\hat{v}(\psi) = 1$	$\hat{v}(\varphi \wedge \psi) = 0$ $\hat{v}(\varphi) = 0 \qquad \hat{v}(\psi) = 0$
$\vee$	$\hat{v}(\varphi \vee \psi) = 0$ $\hat{v}(\varphi) = 0$ $\hat{v}(\psi) = 0$	$\hat{v}(\varphi \vee \psi) = 1$ $\hat{v}(\varphi) = 1 \qquad \hat{v}(\psi) = 1$
$\rightarrow$	$\hat{v}(\varphi \rightarrow \psi) = 0$ $\hat{v}(\varphi) = 1$ $\hat{v}(\psi) = 0$	$\hat{v}(\varphi \rightarrow \psi) = 1$ $\hat{v}(\varphi) = 0 \qquad \hat{v}(\psi) = 1$

As regras estão de acordo com a própria definição dos operadores: para a disjunção entre duas proposições ser falsa, por exemplo, é preciso que ambas as proposições sejam falsas. Enquanto que, para ser verdadeira, basta que apenas uma das proposições seja.

Se, em algum ramo da árvore, encontramos uma das duas configurações a seguir:

$$
\begin{array}{l@{\qquad}l}
\hat{v}(A) = 0 & \hat{v}(A) = 1 \\
\vdots & \vdots \\
\hat{v}(A) = 1 & \hat{v}(A) = 0
\end{array}
$$

Dizemos que o ramo entrou em **contradição** e está **fechado**. Isso porque, em um mesmo mundo, uma variável assume dois valores-verdade, o que é impossível. Tableaux com todos os ramos fechados também são ditos fechados.

3.2.2 O algoritmo

Método dos Tableaux Analíticos

Entrada: um tableau com suas condições iniciais;

Saída: se o tableau estiver fechado, **não**. Se o tableau possuir algum ramo aberto, **sim**.

1. Até que o tableau esteja fechado ou não hajam mais nós possíveis para se aplicar uma regra:
    - 1.1 Aplique uma regra em um vértice do tableau. Marque-o como não aplicável.
    - 1.2 Se algum ramo entrou em contradição com a inserção dos novos nós, marque-o como fechado.

O método consiste em tentarmos aplicar uma regra (e apenas uma vez) em algum do nó da árvore (se possível) até que não sobrem mais nós para aplicar regras ou até que o tableau se torne fechado. **Há uma preferência para a aplicação de regras do tipo α, pois não bifurcam a árvore.**

- $(H \lor (I \lor J)) \to (\neg I \lor G)$ **é refutável?**

Queremos perguntar ao método se existe um mundo possível no qual a proposição é falsa. Desse modo, iniciamos o tableau com a seguinte condição inicial:

$$\hat{v}((G \lor (I \lor J)) \to (\neg I \lor G)) = 0$$

E então, temos:

$\hat{v}((G \lor (I \lor J)) \to (\neg I \lor G)) = 0$ **(I)**

$\hat{v}(G \lor (I \lor J)) = 1$ **(IV)**

$\hat{v}(\neg I \lor G) = 0$ **(II)**

$\hat{v}(\neg I) = 0$ **(III)**

$\hat{v}(G) = 0$

$\hat{v}(I) = 1$

$\hat{v}(G) = 1$ $\qquad$ $\hat{v}(I \lor J) = 1$ **(V)**

$\times$

$\hat{v}(I) = 1$ $\qquad$ $\hat{v}(J) = 1$

O tableau está aberto. Assim, existe um mundo possível para o qual a proposição é falsa. Desse modo, $(H \lor (I \lor J)) \to (\neg I \lor G)$ é refutável.

3.2.3 Complexidade computacional

Vamos agora analisar o custo temporal do método dos Tableaux Analíticos. Sua grande vantagem em relação ao método da Tabela-Verdade é a intuitividade que proporciona, além de ser mais eficiente em alguns casos. Porém, não há uma maneira de identificar que casos são esses. Isso é importante pois, para muitas entradas, o método dos Tableaux Analíticos é ineficiente. Para uma entrada genérica, se supormos que podemos aplicar apenas regras do tipo β, cada nível possuirá o dobro de nós do nível anterior: o método demanda tempo $O(2^n)$, o mesmo do da Tabela-Verdade.

3.3 Método da Resolução

Em 1965, década seguinte à formulação do método dos Tableaux Analíticos, o matemático americano John Alan Robinson publicou um artigo descrevendo um método algorítmico que resolve SAT com 2 propósitos: **eficiente para algumas entradas** e **eficiente em reconhecer essas entradas**. Tal método ficou conhecido como **método da Resolução**. A ideia básica do método consiste na noção de que uma proposição composta por conjunções é insatisfatível se alguma de suas conjunções for insatisfatível. Assim, o método só receberia entradas nesse formato.

3.3.1 Forma Normal Conjuntiva

Uma proposição é dita **literal** se for atômica ou a negação de uma atômica. Uma **cláusula** é uma proposição composta apenas por disjunções de literais. Finalmente, uma proposição na **forma normal conjuntiva** contém apenas conjunções de cláusulas. Toda proposição pode ser transformada para a sua forma normal conjuntiva, como atesta o teorema a seguir.

Teorema 3.3.1

Para toda proposição $\varphi \in PROP$, existe uma proposição $\psi \in PROP$ tal que:

- ψ é logicamente equivalente a φ;
- ψ está na forma normal conjuntiva.

Prova (transformação para a FNC)

Faremos uma prova por indução para mostrar que toda proposição pode ser transformada para uma outra equivalente e na forma normal conjuntiva.

Passo base: φ é atômica.

Temos que $\varphi \equiv \varphi$ e, por definição, φ está na FNC. Assim, podemos tomar $\psi = \varphi$.

Passo indutivo: φ é da forma $(\neg\rho)$.

Hipótese Indutiva: existe ρ tal que $\rho \equiv \varphi$ e ρ está na FNC.

Tese: existe $(\neg\rho)$ tal que $(\neg\rho) \equiv (\neg\psi)$ e $(\neg\rho)$ está na FNC.

Pela hipótese, ρ é uma conjunção de cláusulas. Desse modo, podemos aplicar as equivalências lógicas a seguir para transformar $(\neg\rho)$ para a FNC.

- $\neg\neg\varphi \equiv \varphi$
- $\neg(\varphi \wedge \psi) \equiv (\neg\varphi \vee \neg\psi)$
- $\neg(\varphi \vee \psi) \equiv (\neg\varphi \wedge \neg\psi)$
- $\varphi \vee (\rho \wedge \theta) \equiv (\varphi \vee \rho) \wedge (\varphi \vee \theta)$

Desse modo, está provada a tese.

Passo indutivo: φ é da forma $(\rho \circ \theta)$, onde $\circ \in \{\wedge, \vee, \rightarrow\}$.

Hipótese Indutiva (1): existe ρ' tal que $\rho' \equiv \rho$ e ρ' está na FNC.

Hipótese Indutiva (2): existe θ' tal que $\theta' \equiv \theta$ e θ' está na FNC.

Tese: existe $(\rho' \circ \theta')$ tal que $(\rho' \circ \theta') \equiv (\rho \circ \theta)$ e $(\rho' \circ \theta')$ está na FNC.

Pelas hipóteses, ρ' e θ' estão na FNC. Desse modo, caso $\circ = \wedge$, $(\rho' \circ \theta')$ também está na FNC. Caso $\circ = \vee$ ou $\circ = \rightarrow$, podemos, além das anteriores, aplicar a equivalência lógica a seguir para transformar $(\rho' \circ \theta')$ para a FNC.

- $(\varphi \rightarrow \psi) \equiv (\neg\varphi \vee \psi)$

Desse modo, está provada a tese, e a prova está completa.

3.3.2 Regra da resolução

O método da Resolução possui apenas uma regra, baseada no seguinte princípio:

$$(L_1 \vee L_2) \wedge (\neg L_1 \vee L_3) \equiv (L_1 \vee L_2) \wedge (\neg L_1 \vee L_3) \wedge (L_2 \vee L_3)$$

Ou seja, para quaisquer cláusulas C_1 e C_2, se existir um literal L_1 em C_1 que é complementar a um literal L_2 em C_2 (ou seja, $L_1 = \neg L_2$), podemos construir uma cláusula com os restantes dos literais das cláusulas. Esse processo é dito **propagação** e a cláusula nova é chamada **resolvente** de C_1 e C_2. Se as cláusulas são unitárias (possuem apenas um literal), o resolvente delas, se existir, é a **cláusula vazia**, sem literais. A cláusula vazia representa uma contradição e, portanto, insatisfatibilidade.

Método da Resolução

Entrada: uma proposição φ na forma normal conjuntiva;

Saída: se a cláusula vazia for encontrada, **sim**. Caso contrário, **não**.

1. Até que a cláusula vazia seja encontrada ou todos os pares de cláusulas possíveis foram propagados:

   1.1 Para cada par de cláusulas de φ:

   1.2 Se o par contiver literais complementares, aplique a regra da resolução.

   1.3 Remova literais repetidos do resolvente e adicione-o ao conjunto de cláusulas.

Apesar de o algoritmo propagar pares de cláusulas quaisquer, **há uma preferência para que**

pelo menos uma cláusula do par seja unitária, pois possibilita uma "redução de tamanho" da outra.

- $(x \vee y) \wedge (\neg x \vee y) \wedge (x \vee \neg y) \wedge (\neg x \vee \neg y)$ **é insatisfatível?**

Queremos saber se é possível encontrar a cláusula vazia a partir do conjunto de cláusulas da proposição. Assim:

$$(x \vee y)^{C_1} \wedge (\neg x \vee y)^{C_2} \wedge (x \vee \neg y)^{C_3} \wedge (\neg x \vee \neg y)^{C_4}$$

C_1 e $C_2 \Rightarrow y^{C_5}$

C_5 e $C_3 \Rightarrow x^{C_6}$

C_6 e $C_4 \Rightarrow (\neg y)^{C_7}$

C_7 e $C_5 \Rightarrow ()$

A cláusula vazia foi encontrada. Desse modo, $(x \vee y) \wedge (\neg x \vee y) \wedge (x \vee \neg y) \wedge (\neg x \vee \neg y)$ é insatisfatível.

3.3.3 Cláusulas de Horn

Nos anos 1950, o matemático também americano Alfred Horn estudou intensamente as propriedades de um determinado tipo de proposição que tinha um certo apelo à representação de "condições implicam em uma consequência":

$$\begin{aligned}&(x_1 \wedge x_2 \wedge x_3 \wedge ... \wedge x_n) \rightarrow y\\ &\equiv \neg(x_1 \wedge x_2 \wedge x_3 \wedge ... \wedge x_n) \vee y\\ &\equiv (\neg x_1 \vee \neg x_2 \vee \neg x_3 \vee ... \vee \neg x_n \vee y)\end{aligned}$$

Cláusulas nesse formato, **que contém apenas um literal positivo**, são chamadas **cláusulas de Horn**, em homenagem ao matemático.

É possível construir outro algoritmo, similar ao método da Resolução, quando a proposição a ser avaliada possui apenas cláusulas de Horn, que dá ênfase apenas a cláusulas unitárias.

Propagação da Cláusula Unitária

Entrada: uma proposição φ na forma normal conjuntiva contendo apenas cláusulas de Horn;

Saída: se a cláusula vazia for encontrada, **sim**. Caso contrário, **não**.

1. Para cada cláusula unitária L de φ:
   - 1.1 Para cada cláusula C, exceto L, de φ:
   - 1.2 Se C contém L, remova C do conjunto de cláusulas.
   - 1.3 Se C contém $\neg L$, remova $\neg L$ de C e remova os literais repetidos de C.
     - 1.3.1 Se o resolvente for uma cláusula unitária, adicione-o ao conjunto de cláusulas unitárias.
     - 1.3.2 Caso seja uma cláusula vazia, retorne **sim, é insatisfatível**.
2. Se a cláusula vazia não foi encontrada, retorne **não é insatisfatível**.

3.3.4 Complexidade computacional

Foi provado que o custo do método da Resolução é da ordem de $O(2^n)$, o mesmo do da Tabela-Verdade. Porém, com um diferencial importante: quando o método recebe como entrada uma proposição contendo apenas cláusulas de Horn, a propagação da cláusula unitária permite ao método rodar em $O(n)$, ou seja, linear. Por esse motivo, o método da Resolução com cláusulas de Horn é a base para a programação lógica, como Prolog.

3.4 Método da Dedução Natural

Em meados de 1934, o matemático alemão Gerhard Gentzen buscou definir um método de formalização dos procedimentos dedutivos utilizados naturalmente pelos matemáticos em suas demonstrações. Assim, buscou encontrar um conjunto de regras simples de deduções que correspondesse ao conjunto de passos dedutivos em provas matemáticas. Ao invés de partir do conceito de valor-verdade, seu método partiu da noção de **regra de dedução**, que independe de uma interpretação em valores booleanos. Seu método ficou conhecido como **método da Dedução Natural**.

Antes de detalhar mais o método, pode ser útil relembrar os formatos de provas matemáticas. Em particular, vejamos uma a seguir.

Teorema 3.4.1

Se x e y são naturais pares, então $x + y$ é par.

Prova

Queremos provar que a soma de dois naturais pares também é par. Temos:

1. Suponha que x e y são pares.
2. Por (1), $x = 2a$ para algum $a \in \mathbb{N}$.
3. Por (1), $y = 2b$ para algum $b \in \mathbb{N}$.
4. Por (2) e (3), $x + y = 2a + 2b = 2(a + b)$.
5. Desse modo, $x + y$ é par.

Observe que o teorema é uma implicação: **se** [**condição**] **então** [**conclusão**]. Desse modo, para prová-lo, devemos mostrar que sempre que a condição é satisfeita, a conclusão também é. Tomando isso como base, a primeira linha da prova representa a **suposição** ou **premissa** da condição. As 3 seguintes são passos intermediários até conseguirmos deduzir a quinta linha, que é justamente a conclusão. Desse modo, deduzimos a implicação. Passos como esses foram sintetizados por Gentzen em um conjunto de regras simples de dedução, que veremos a seguir.

3.4.1 Regras de dedução

Similar ao método dos Tableaux Analíticos, Gentzen definiu dois tipos de regras de dedução, contemplando duas regras por operador lógico.

Regras de Introdução

São regras que determinam as condições mínimas para que se possa deduzir uma proposição cujo operador principal fosse o em questão.

$$\begin{array}{c|c|c}
\bigwedge & \bigvee & \longrightarrow \\
\dfrac{\varphi \quad \psi}{\varphi \wedge \psi} & \dfrac{\varphi}{\varphi \vee \psi} \qquad \dfrac{\psi}{\varphi \vee \psi} & \begin{array}{c}[\varphi]\\ \vdots \\ \psi \\ \hline \varphi \to \psi\end{array}
\end{array}$$

A 3ª regra é chamada **modus ponens** e corresponde à prova que fizemos no início. Uma proposição entre colchetes, como $[\varphi]$, simboliza uma **premissa adicional**.

Regras de Eliminação

São regras que determinam as consequências imediatas que podem ser obtidas a partir de uma proposição cujo operador principal fosse o em questão.

$$
\begin{array}{c|c|c}
\bigwedge & \bigvee & \to \\
\dfrac{\varphi \wedge \psi}{\varphi} \quad \dfrac{\varphi \wedge \psi}{\psi} & \dfrac{\varphi \vee \psi \quad \begin{array}{c}[\varphi]\\ \vdots \\ \theta\end{array} \quad \begin{array}{c}[\psi]\\ \vdots \\ \theta\end{array}}{\theta} & \dfrac{\varphi \quad \varphi \to \psi}{\psi}
\end{array}
$$

A 2ª regra corresponde a uma **prova por casos**: caso ambos os casos levem a uma mesma conclusão, podemos deduzí-la.

Regras que utilizam de premissas adicionais, como a introdução da implicação e a eliminação da disjunção, as **descartam** ao serem aplicadas. Premissas descartadas não podem mais ser usadas no restante do método.

3.4.2 Dedutibilidade

Em contrapartida aos métodos anteriores, que aceitam as instâncias de satisfatilibidade que vimos no capítulo anterior, uma vez que o método da Dedução Natural ignora a noção de valoração-verdade, ele procura responder a outra pergunta: "φ é dedutível?".

Dedutibilidade

Seja φ uma proposição e Γ um conjunto de proposições (chamadas de premissas).

- φ é **dedutível** a partir de Γ ($\Gamma \vdash \varphi$ se existe uma árvore de dedução cuja raiz é φ, cujas folhas são premissas de Γ ou premissas adicionais devidamente descartadas e cujos vértices internos são aplicações das regras de dedução.

Podemos relacionar dedutibilidade e satisfatilibidade a partir do seguinte teorema, que não demonstraremos:

Teorema 3.4.2: Teorema da Completude-Corretude

Seja Γ um conjunto de proposições e φ uma proposição.

$$\Gamma \vdash \varphi \text{ se, e somente se, } \Gamma \vDash \varphi$$

- $\{P \to Q, Q \to R\} \vdash P \to (Q \wedge R)$?

Queremos determinar se existe uma árvore de dedução na qual $P \to (Q \wedge R)$ é a raiz, $P \to Q$ e $Q \to R$ são suas folhas e seus vértices são aplicações das regras de dedução. Inicialmente, começamos a dedução natural de baixo para cima, para obter premissas adicionais e analisar o formato da árvore. Desse modo, obtemos:

$$
\dfrac{\dfrac{Q \qquad \begin{array}{c}[P]\\ \vdots \\ R\end{array}}{Q \wedge R}\, I\wedge}{P \to (Q \wedge R)}\, I\!\to
$$

Obtivemos $[P]$ como premissa adicional. Além disso, a árvore nos diz que devemos deduzir ambos Q e R do conjunto de premissas. Assim, montamos agora a árvore de cima para baixo, para revelar o restante.

$$\begin{array}{c} [P] \quad P \to Q \quad Q \to R \\ \vdots \\ \dfrac{\dfrac{Q \qquad\qquad\qquad R}{Q \wedge R}\; I\wedge}{P \to (Q \wedge R)}\; I{\to} \end{array} \quad \rightsquigarrow \quad \dfrac{\dfrac{Q \qquad \dfrac{\dfrac{\cancel{[P]}^{1} \quad P \to Q}{Q}\; E{\to} \qquad Q \to R}{R}\; E{\to}}{Q \wedge R}\; I\wedge}{P \to (Q \wedge R)}\; I{\to}\ (1)$$

Podemos descrever nosso raciocínio da seguinte forma:

1. Para deduzir $P \to (Q \wedge R)$, precisamos supor P e deduzir $(Q \wedge R)$ a partir da suposição.
2. Por sua vez, para deduzirmos $(Q \wedge R)$, precisamos deduzir ambos Q e R:
3. A partir das premissas $[P]$ e $P \to Q$, podemos usar a eliminação da implicação para deduzir Q.
4. E então, a partir de Q e $Q \to R$, novamente usamos a eliminação da implicação para deduzir R.
5. Finalizamos então usando a introdução da conjunção em Q e R e, como alcançamos $Q \wedge R$ a partir de $[P]$, podemos deduzir $P \to (Q \wedge R)$. Ao ser aplicada, a regra de introdução da implicação descarta $[P]$.

Como tal árvore de dedução existe, temos que $P \to (Q \wedge R)$ é dedutível a partir de $\{P \to Q, Q \to R\}$.

3.4.3 Negação

Você já deve ter percebido que a negação não foi introduzida nas regras de dedução. Isso se deve ao fato de que esse operador não possui regras próprias. Porém, é importante notar que:

$$(\neg\varphi) \equiv (\neg\varphi \vee \bot) \equiv (\varphi \to \bot)$$

Desse modo, usamos as regras da implicação para lidar com a negação. Porém, ao fazermos isso, devemos lidar com alguns problemas que surgem. Por exemplo, ao tentarmos construir uma árvore de prova para responder a "$\{(P \vee Q), \neg P\} \vdash Q$?", temos:

$$\begin{array}{c} (P \vee Q) \quad (P \to \bot) \\ \vdots \\ Q \end{array} \quad \rightsquigarrow \quad \dfrac{P \vee Q \qquad \dfrac{[P] \quad P \to \bot}{\bot}\; E{\to} \qquad [Q]}{}$$

A eliminação da disjunção não é concluída pois as conclusões de ambos os casos foram diferentes: $\bot$ e $[Q]$. Como não podemos aplicar mais nenhuma regra, simbolizando que a árvore de prova não existe, o método responde não. Porém, a Tabela-Verdade diz o contrário:

P	Q	$\neg P$	$P \vee Q$
0	0	1	0
0	**1**	**1**	**1**
1	0	0	1
1	1	0	1

Para preservar a corretude do método da Dedução Natural, devemos introduzir mais duas regras de dedução:

$$\dfrac{\bot}{\varphi} \ \text{se } \varphi \text{ for atômica} \quad \Bigg| \quad \begin{array}{c} [\neg\varphi] \\ \vdots \\ \dfrac{\bot}{\varphi} \end{array}$$

A primeira regra se chama **Princípio da Explosão** ou **Ex Falsum Quodlibet**. Ela afirma que *do falso, é possível deduzir qualquer coisa*. A segunda se chama **Redução ao Absurdo** ou **Reductio ad Absurdum** e é equivalente a provas por contradição.

A introdução de tais regras é importante o suficiente para dividir a lógica em três vertentes:

Lógica Minimal: admite apenas o conjunto de regras de dedução básico em suas demonstrações;

Lógica Intuicionista: admite, além do conjunto de regras de dedução básico, o princípio da explosão em suas demonstrações;

Lógica Clássica: admite, além do conjunto de regras de dedução básico, o princípio da explosão e a redução ao absurdo em suas demonstrações.

Ainda, com a regra de redução ao absurdo somos capazes de provar duas das três **leis clássicas do pensamento**, que não conseguíamos anteriormente:

Lei do Terceiro Excluído: $(\neg\varphi \vee \varphi)$ é tautologia.

Lei da Dupla Negação: $(\neg\neg\varphi) \equiv \varphi$

Desse modo, também podemos dizer que a Lógica Clássica reconhece as duas leis.

3.4.4 O algoritmo

Apesar de um algoritmo para a Dedução Natural existir, o método possui uma intuitividade muito alta quando o executamos do mesmo jeito que pensamos "naturalmente". Desse modo, o algoritmo, que usa o conceito de *backtracking*, está a seguir só para fins de consulta.

Método da Dedução Natural

Entrada: uma proposição φ e um conjunto de premissas;

Saída: se uma árvore de dedução for construída corretamente, **sim**. Caso contrário, **não**.

1. Aplique o contrário das regras de introdução em φ até que não seja mais possível e adicione as premissas adicionais obtidas ao conjunto de premissas.
2. Execute o seguinte procedimento recursivo:

   2.1 Se uma regra de dedução pode ser aplicada na árvore de dedução, aplique-a. Caso contrário, retorne **não**.

   2.2 Se a regra aplicada foi a introdução da implicação, eliminação da disjunção ou redução ao absurdo e uma premissa adicional foi usada, remova-a do conjunto de premissas.

   2.3 Repita o processo recursivamente.

   2.4 Se o resultado da recursão foi não e árvore de prova está formada, retorne **sim**. Caso contrário, se puder, refaça recursivamente a escolha de aplicação da regra. Se não puder, retorne **não**.

3.4.5 Forma Normal

Além de podermos aplicar o método da Dedução Natural do mesmo jeito que pensamos (daí o nome "natural"), uma outra característica positiva do método é a capacidade de avaliar matematicamente, dentre as soluções possíveis para um problema, a melhor. Eventualmente, em uma prova de Dedução Natural, podemos encontrar situações como as seguir:

$$\dfrac{\dfrac{\varphi \quad \psi}{\varphi \wedge \psi}\, I\wedge}{\varphi}\, E\wedge \qquad\qquad \dfrac{\dfrac{\varphi \wedge \psi}{\varphi}\, E\wedge \quad \dfrac{\varphi \wedge \psi}{\psi}\, E\wedge}{\varphi \wedge \psi}\, I\wedge$$

Na primeira situação, uma vez que já temos φ como vértice da árvore, não precisamos deduzí-la novamente. O mesmo ocorre com $\varphi \wedge \psi$ na segunda situação. Essas operações desnecessárias são chamadas **redundâncias** e ocorrem ao aplicar as regras de introdução e eliminação em sequência para o mesmo operador.

Podemos classificar redundâncias em dois tipos: o tipo β, que consiste na aplicação de uma regra de introdução seguida de uma de eliminação para o mesmo operador; e o tipo η, que consiste na aplicação de uma regra de eliminação seguida de uma de introdução para o mesmo operador.

Regras de Redução

Uma árvore de dedução que não contém redundâncias é dita estar na **forma normal**. Podemos transformar qualquer árvore de prova para a sua forma normal, como provou o lógico sueco Dag Prawitz em 1965:

Teorema 3.4.3: Teorema da Normalização

Seja φ uma proposição. Se $\vdash \varphi$, então existe uma árvore de prova normal para φ.

Para provar seu teorema, Prawitz definiu um conjunto de 6 regras (3 para cada tipo de redundância) que nos permitem eliminar redundâncias de uma árvore de dedução, chamadas **regras de redução**. Caso a redundância seja do tipo β, a proposição no centro da redução é dita **máxima**. Caso seja do tipo η, a proposição é dita **mínima**.

Redução para redundâncias do tipo β

Operador	Redundância	Redução
$\bigwedge$	$\begin{matrix}\dfrac{\dfrac{\begin{matrix}\nabla_1\\ \varphi\end{matrix}\quad\begin{matrix}\nabla_2\\ \psi\end{matrix}}{\varphi\wedge\psi}\,I\wedge}{\varphi \text{ ou } \psi}\,E\wedge\\ \Delta_3\end{matrix}$	$\begin{matrix}\nabla_1\\ \varphi \text{ ou } \psi\\ \Delta_3\end{matrix}$
$\bigvee$	$\begin{matrix}\dfrac{\dfrac{\begin{matrix}\nabla_1\\ \varphi \text{ ou } \psi\end{matrix}}{\varphi\vee\psi}\,I\vee\quad\begin{matrix}[\varphi]\\ \Delta_2\\ \theta\end{matrix}\quad\begin{matrix}[\psi]\\ \Delta_3\\ \theta\end{matrix}}{\theta}\,E\vee\\ \Delta_4\end{matrix}$	$\begin{matrix}\nabla_1\\ \varphi \text{ ou } \psi\\ \Delta_2 \text{ ou } \Delta_3\\ \theta\\ \Delta_4\end{matrix}$
$\longrightarrow$	$\begin{matrix}\dfrac{\begin{matrix}\nabla_1\\ \varphi\end{matrix}\quad\dfrac{\begin{matrix}[\varphi]\\ \Delta_2\\ \psi\end{matrix}}{\varphi\to\psi}\,I\!\to}{\psi}\,E\!\to\\ \Delta_3\end{matrix}$	$\begin{matrix}\nabla_1\\ \varphi\\ \Delta_2\\ \psi\\ \Delta_3\end{matrix}$

Redução para redundâncias do tipo η

Operador	Redundância	Redução
$\bigwedge$	$\begin{array}{c}\dfrac{\dfrac{\begin{array}{c}\nabla_1\\ \varphi\wedge\psi\end{array}}{\varphi}E\wedge \quad \dfrac{\begin{array}{c}\nabla_1\\ \varphi\wedge\psi\end{array}}{\psi}E\wedge}{\varphi\wedge\psi}I\wedge \\ \Delta_2\end{array}$	$\begin{array}{c}\nabla_1\\ \varphi\wedge\psi\\ \Delta_2\end{array}$
$\bigvee$	$\begin{array}{c}\dfrac{\begin{array}{c}\nabla_1\\ \varphi\vee\psi\end{array} \quad \dfrac{[\varphi]}{\varphi\vee\psi}I\vee \quad \dfrac{[\psi]}{\varphi\vee\psi}I\vee}{\varphi\vee\psi}E\vee \\ \Delta_2\end{array}$	$\begin{array}{c}\nabla_1\\ \varphi\vee\psi\\ \Delta_2\end{array}$
$\longrightarrow$	$\begin{array}{c}\dfrac{\dfrac{[\varphi] \quad \begin{array}{c}\nabla_1\\ \varphi\to\psi\end{array}}{\psi}E\to}{\varphi\to\psi}I\to \\ \Delta_2\end{array}$	$\begin{array}{c}\nabla_1\\ \varphi\to\psi\\ \Delta_2\end{array}$

3.5 Método do Cálculo de Sequentes

De modo a estudar propriedades sobre o método da Dedução Natural, Gentzen definiu uma ferramenta algorítmica que lidava com operações sobre sequentes: estruturas que são equivalentes a provas. Ocorre que a ferramenta se tornou mais um poderoso e útil método algorítmico que resolve SAT, até mesmo para contornar alguns dos problemas que a Dedução Natural possuía. O método foi chamado de **método do Cálculo de Sequentes**.

Um **sequente** é uma estrutura do tipo

$$\varphi_1, \varphi_2, \varphi_3, ..., \varphi_n \vdash \psi_1, \psi_2, \psi_3, ..., \psi_m$$

Essa estrutura é equivalente a $(\varphi_1 \wedge \varphi_2 \wedge \varphi_3 \wedge ... \wedge \varphi_n) \to (\psi_1 \vee \psi_2 \vee \psi_3 \vee ... \vee \psi_m)$. A parte à esquerda contém as **premissas** do sequente, enquanto a parte à direita contém as **conclusões** do sequente.

3.5.1 Regras de dedução

O método contém um conjunto considerável de regras em comparação aos outros métodos, divididos em 5 categorias.

Axioma

$$\dfrac{}{A \vdash A}$$

Um axioma é a "unidade básica" do método do Cálculo de Sequentes: todas as folhas devem ser axiomas para o método responder positivamente.

Regra do Corte

$$\dfrac{\Gamma \vdash A, \Delta \qquad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

A regra do corte é a generalização do *modus ponens* clássico: atesta que, se uma proposição A aparece como conclusão de uma prova e em outra aparece como premissa, então podemos criar uma prova sem a aparição de A.

Contudo, foi provado por Gentzen em 1934 que, apesar de ser útil em algumas situações, a regra do corte para o cálculo de sequentes é sempre dispensável:

Teorema 3.5.1: Teorema da Eliminação do Corte

Qualquer árvore de dedução em cálculo de sequentes que faz uso da regra do corte possui uma equivalente sem o uso da regra.

Regras de Introdução à Esquerda

São regras que são aplicáveis à esquerda da catraca ($\vdash$).

$$
\begin{array}{c|c|c|c}
\neg & \bigwedge & \bigvee & \longrightarrow \\
\dfrac{\Gamma \vdash \varphi, \Delta}{\Gamma, \neg\varphi \vdash \Delta} & \dfrac{\Gamma, \varphi, \psi \vdash \Delta}{\Gamma, \varphi \wedge \psi \vdash \Delta} & \dfrac{\Gamma, \varphi \vdash \Delta \qquad \Sigma, \psi \vdash \Pi}{\Gamma, \Sigma, \varphi \vee \psi \vdash \Delta, \Pi} & \dfrac{\Gamma \vdash \varphi, \Delta \qquad \Sigma, \psi \vdash \Pi}{\Gamma, \Sigma, \varphi \to \psi \vdash \Delta, \Pi}
\end{array}
$$

Regras de Introdução à Direita

São regras que são aplicáveis à direita da catraca ($\vdash$).

$$
\begin{array}{c|c|c|c}
\neg & \bigwedge & \bigvee & \longrightarrow \\
\dfrac{\Gamma, \varphi \vdash \Delta}{\Gamma \vdash \neg\varphi, \Delta} & \dfrac{\Gamma \vdash \varphi, \Delta \qquad \Sigma \vdash \psi, \Pi}{\Gamma, \Sigma \vdash \varphi \wedge \psi, \Delta, \Pi} & \dfrac{\Gamma \vdash \varphi, \psi, \Delta}{\Gamma \vdash \varphi \vee \psi, \Delta} & \dfrac{\Gamma, \varphi \vdash \psi, \Delta}{\Gamma \vdash \varphi \to \psi, \Delta}
\end{array}
$$

Regras Estruturais

São regras que permitem modificar a estrutura de um sequente, obedecendo equivalências lógicas. Elas podem ser aplicadas tanto à esquerda quanto à direita da catraca.

$$
\begin{array}{c|c|c}
\textsc{Permutação} & \textsc{Contração} & \textsc{Enfraquecimento} \\
\dfrac{\Gamma_1, \varphi, \psi, \Gamma_2 \vdash \Delta}{\Gamma_1, \psi, \varphi, \Gamma_2 \vdash \Delta} & \dfrac{\Gamma, \varphi, \varphi \vdash \Delta}{\Gamma, \varphi, \vdash \Delta} & \dfrac{\Gamma \vdash \Delta}{\Gamma, A, \vdash \Delta} \\
\\
\dfrac{\Gamma \vdash \Delta_1, \psi, \varphi, \Delta_2}{\Gamma \vdash \Delta_1, \varphi, \psi, \Delta_2} & \dfrac{\Gamma \vdash \varphi, \varphi, \Delta}{\Gamma \vdash \varphi, \Delta} & \dfrac{\Gamma \vdash \Delta}{\Gamma \vdash A, \Delta}
\end{array}
$$

De baixo para cima, o método do Cálculo de Sequentes se comporta isomorficamente ao método dos Tableaux Analíticos, se considerarmos premissas como tendo valor-verdade verdadeiro e conclusões como tendo valor-verdade falso. Desse modo, é útil montar a árvore de baixo para cima. E, de modo análogo, **há uma preferência para a aplicação de regras de introdução que não bifurquem a árvore**, mesmo que regras estruturais devam ser usadas para esse fim.

Método do Cálculo de Sequentes

Entrada: um sequente;

Saída: se uma árvore de dedução for construída corretamente, **sim**. Caso contrário, **não**.

1. Se uma regra de dedução pode ser aplicada ao lado da catraca, aplique-a. Caso contrário, retorne **não**.
2. Repita o processo recursivamente.
3. Se o resultado da recursão foi não e árvore de prova está formada, ou seja, a raiz é o sequente original e todas as folhas são axiomas, retorne **sim**. Caso contrário, se puder, refaça recursivamente a escolha de aplicação da regra. Se não puder, retorne **não**.

- $\vdash A \vee \neg A$**?**

Queremos perguntar se existe uma árvore de prova na qual $\vdash A \vee \neg A$ é a raiz, e suas folhas são axiomas. Temos:

$$\dfrac{\dfrac{\dfrac{\overline{A \vdash A}}{\vdash \neg A, A}\ \neg D}{\vdash A, \neg A}\ PER}{\vdash A \vee \neg A}\ \vee D$$

Como tal árvore de prova existe, temos que $\vdash A \vee \neg A$.

Exercícios Resolvidos

1. Dada as sentenças a seguir:

   1: Se você usa Linux e usa Mozilla como navegador, evita problemas.

   2: Se você usa Internet Explorer como navegador, terá problemas.

   3: Você usa Mozilla como navegador.

   4: Você usa Internet Explorer como navegador.

   5: Você não usa Linux.

   Prove que $\{1, 2, 3, 4\} \vDash 5$, usando:

   (a) Método dos Tableaux Analíticos

   (b) Método da Resolução

   (c) Método da Dedução Natural

   (d) Método do Cálculo de Sequentes

   Antes de tudo, precisamos escrever as sentenças na linguagem da Lógica Proposicional. Definindo as sentenças atômicas:

   - L: você usa Linux;
   - M: você usa Mozilla como navegador;
   - I: você usa Internet Explorer como navegador;
   - P: você terá problemas;

   Assim:

   1: $(L \wedge M) \rightarrow \neg P$

   2: $I \rightarrow P$

   3: M

   4: I

   5: $\neg L$

   Desse modo, queremos provar que $\{(L \wedge M) \rightarrow \neg P, I \rightarrow P, M, I\} \vDash \neg L$.

   (a) **Método dos Tableaux Analíticos**
   A definição de "φ consequência lógica de Γ" é que toda valoração-verdade que torna todas as sentenças de Γ verdadeiras, também torna φ verdadeira. Porém, Tableaux Analíticos não responde "para toda valoração", e sim "existe uma valoração". Desse modo, queremos perguntar se existe uma valoração (mundo possível) na qual as sentenças de Γ são verdadeiras, mas φ é falsa, e esperamos que o método negue.

   Temos:

$\hat{v}((L \land M) \to \neg P) = 1$ **(III)**

$\hat{v}(I \to P) = 1$ **(II)**

$\hat{v}(M) = 1$

$\hat{v}(I) = 1$

$\hat{v}(\neg L) = 0$ **(I)**

$\hat{v}(L) = 1$

$\hat{v}(I) = 0$ $\qquad$ $\hat{v}(P) = 1$

$\times$

(V) $\hat{v}(L \land M) = 0$ $\qquad$ $\hat{v}(\neg P) = 1$ **(IV)**

$\hat{v}(L) = 0$ $\qquad$ $\hat{v}(M) = 0$ $\qquad$ $\hat{v}(P) = 0$

$\times$ $\qquad$ $\times$ $\qquad$ $\times$

O tableau está fechado: não existe mundo possível no qual $\neg L$ é falsa, enquanto as proposições de $\{(L \land M) \to \neg P, I \to P, M, I\}$ são verdadeiras. Desse modo, qualquer valoração-verdade que satisfaz o conjunto deve satisfazer também $\neg L$ e, portanto, $\neg L$ é consequência lógica de $\{(L \land M) \to \neg P, I \to P, M, I\}$.

(b) **Método da Resolução**

O método da Resolução não responde se uma proposição é consequência lógica de um conjunto de sentenças. Desse modo, usaremos a seguinte equivalência lógica:

$$\Gamma \models \varphi \text{ se, e somente se, } \Gamma \cup \{\neg\varphi\} \text{ é insatisfatível}$$

Agora, precisamos converter todas as sentenças de $\Gamma \cup \{\neg\varphi\}$ para as suas respectivas formas normais conjuntivas.

- $(L \land M) \to \neg P$
  $\equiv \neg(L \land M) \lor \neg P$
  $\equiv \neg L \lor \neg M \lor \neg P$ **FNC**
- $I \to P$
  $\equiv \neg I \lor P$ **FNC**
- I, M, L **FNC**

Assim, queremos perguntar ao método se é possível encontrar a cláusula vazia a partir do conjunto de cláusulas que possuímos. Assim:

$$(\neg L \lor \neg M \lor \neg P)^{C_1} \land (\neg I \lor P)^{C_2} \land I^{C_3} \land M^{C_4} \land L^{C_5}$$

C_1 e $C_4 \Rightarrow (\neg L \lor \neg P)^{C_6}$

C_2 e $C_3 \Rightarrow P^{C_7}$

C_6 e $C_7 \Rightarrow (\neg L)^{C_8}$

C_8 e $C_5 \Rightarrow ()$

A cláusula vazia foi encontrada. Desse modo, $\{(L \land M) \to \neg P, I \to P, M, I, L\}$ é insatisfatível e, portanto, $\neg L$ é consequência lógica de $\{(L \land M) \to \neg P, I \to P, M, I\}$.

(c) **Método da Dedução Natural**
O método da Dedução Natural não responde se uma proposição é consequência lógica de um conjunto de premissas, mas responde algo bem parecido:

$$\Gamma \vDash \varphi \text{ se, e somente se, } \Gamma \vdash \varphi$$

Desse modo, queremos perguntar se existe uma árvore de dedução onde $\neg L$ é a raiz, $(L \wedge M) \to \neg P$, $I \to P$, M e I são folhas além de premissas adicionais devidamente descartadas e os vértices da árvore são aplicações das regras de dedução. Temos:

$$\dfrac{\begin{array}{c}[L]\\ \vdots \\ \bot\end{array}}{L \to \bot}\,I\!\to \quad \rightsquigarrow \quad \dfrac{\dfrac{\dfrac{I \quad I\to P}{P}\,E\!\to \qquad \dfrac{\dfrac{[\cancel{L}]^{1} \quad M}{L\wedge M}\,I\wedge \qquad (L\wedge M)\to(P\to\bot)}{P\to\bot}\,E\!\to}{\bot}\,E\!\to}{L\to\bot}\,I\!\to (1)$$

Tal árvore de dedução existe. Desse modo, $\neg L$ é dedutível a partir de $\{(L\wedge M) \to \neg P, I \to P, M, I\}$ e, portanto, também é consequência lógica do mesmo.

(d) **Método do Cálculo de Sequentes**
Similar ao método da Dedução Natural, usaremos a mesma equivalência lógica para transformar a entrada em algo que o método do Cálculo de Sequentes possa responder:

$$\Gamma \vDash \varphi \text{ se, e somente se, } \Gamma \vdash \varphi$$

E assim, queremos perguntar se existe uma árvore de dedução na qual $(L\wedge M) \to \neg P, I \to P, M, I \vdash \neg L$ seja sua raiz e suas folhas sejam axiomas. Usaremos linhas duplas para sinalizar uma permutação feita mais de uma vez. Temos:

$$\dfrac{\begin{array}{c}\dfrac{\begin{array}{c}\dfrac{\begin{array}{c}\dfrac{\dfrac{\dfrac{}{L\vdash L} \quad \dfrac{}{M\vdash M}}{M,L\vdash L\wedge M}\,D\wedge \qquad \dfrac{\dfrac{}{P\vdash P}}{P,\neg P\vdash}\,E\neg}{M,L,P,(L\wedge M)\to\neg P\vdash}\,E\!\to \\ \hline\hline (L\wedge M)\to\neg P, M, L, P\vdash \end{array}\,PER \qquad \dfrac{}{I\vdash I}}{(L\wedge M)\to\neg P, M, I, L, I\to P\vdash}\,E\!\to \\ \hline\hline (L\wedge M)\to\neg P, I\to P, M, I, L\vdash \end{array}\,PER}{} \end{array}}{(L\wedge M)\to\neg P, I\to P, M, I\vdash\neg L}\,D\neg$$

Tal árvore de dedução existe. Desse modo, $\neg L$ é dedutível a partir de $\{(L\wedge M) \to \neg P, I \to P, M, I\}$ e, portanto, também é consequência lógica do mesmo.

2. Prove que $((P \to Q) \to P) \to P$ é tautologia, usando:
   (a) Método dos Tableaux Analíticos
   (b) Método da Resolução
   (c) Método da Dedução Natural
   (d) Método do Cálculo de Sequentes

(a) **Método dos Tableaux Analíticos**
Tableaux Analíticos não responde se toda valoração-verdade satisfaz a proposição. Desse modo, iremos perguntar se existe uma valoração-verdade que a refuta, e

esperamos que o método negue. Temos:

$\hat{v}(((P \to Q) \to P) \to P) = 0$ **(I)**

$\hat{v}((P \to Q) \to P) = 1$ **(II)**

$\hat{v}(P) = 0$

(III) $\hat{v}(P \to Q) = 0$ | $\hat{v}(P) = 1$

$\times$

$\hat{v}(P) = 1$

$\hat{v}(Q) = 0$

$\times$

O tableau está fechado: não existe mundo possível no qual $((P \to Q) \to P) \to P$ é falsa, e, portanto, é tautologia.

(b) **Método da Resolução**
O método da Resolução não responde se uma proposição é tautologia. Desse modo, usaremos a equivalência lógica:

$$\varphi \text{ é tautologia se, e somente se, } \neg\varphi \text{ é insatisfatível}$$

Assim, transformando a proposição para a FNC:

- $\neg(((P \to Q) \to P) \to P)$
  $\equiv \neg(\neg((P \to Q) \to P) \vee P)$
  $\equiv \neg(\neg(\neg(P \to Q) \vee P) \vee P)$
  $\equiv \neg(\neg(\neg(\neg P \vee Q) \vee P) \vee P)$
  $\equiv \neg\neg(\neg(\neg P \vee Q) \vee P) \wedge \neg P$
  $\equiv (\neg(\neg P \vee Q) \vee P) \wedge \neg P$
  $\equiv ((P \wedge \neg Q) \vee P) \wedge \neg P$
  $\equiv P \wedge (P \wedge \neg Q) \wedge \neg P$

II

Lógica de Primeira Ordem

Enquanto a Lógica Proposicional tem como base proposições, onde enunciados são inteiramente representados por variáveis, Gottlob Frege buscou, em 1879, obter uma linguagem simbólica mais rica, que representa enunciados na qual os objetos mencionados nesses enunciados tenham uma representação própria. Observe as seguintes sentenças declarativas abaixo.

1. Se o unicórnio é lenda, é imortal, mas se não é lenda, é mamífero.
2. O unicórnio, se é imortal ou mamífero, é chifrudo.
3. O unicórnio, se é chifrudo, é bruxaria.

Queremos saber: o unicórnio é lenda? É bruxaria? É chifrudo? Vamos representar as sentenças na lógica proposicional na seguinte forma:

- l = "O unicórnio é lenda."
- i = "O unicórnio é imortal."
- m = "O unicórnio é mamífero."
- c = "O unicórnio é chifrudo."
- b = "O unicórnio é bruxaria."

1. $(l \to i) \land (\neg l \to m)$
2. $(i \lor m) \to c$
3. $c \to b$

Basta saber então, se $\{1, 2, 3\}$ acarreta em l, c ou b. Podemos usar algum dos métodos algorítmicos que resolvem SAT para resolver esse problema. Agora, vejamos as sentenças abaixo.

4. O jumento é primo do unicórnio.
5. Todo primo do unicórnio é chifrudo.
6. Algum primo do unicórnio não é bruxaria.
7. A fêmea do jumento é chifruda.

Na lógica proposicional, cada uma das sentenças tem que ser representada por uma variável. Isso implica em perda de expressividade, pois não podemos representar conceitos como "primo de", "todo", "algum", "fêmea de". Sendo assim, queremos usar símbolos que nos permitam representar os objetos e as relações entre eles. Temos:

Objetos j: jumento; u: unicórnio; $f(j)$: fêmea do jumento

Predicados e relações $L(x)$: x é lenda; $I(x)$: x é imortal; $M(x)$: x é mamífero; $C(x)$: x é chifrudo; $B(x)$: x é bruxaria; $P(x, y)$: x é primo de y

Adicionalmente, usamos os símbolos $\forall x$ para representar "para todo x" e $\exists x$ para "existe x". Assim, podemos representar as sentenças 1 a 7 como:

1. $(L(u) \to I(u)) \land (\neg L(u) \to M(u))$
2. $(I(u) \lor M(u)) \to C(u)$
3. $C(u) \to B(u)$
4. $P(j, u)$
5. $\forall x(P(x, u) \to C(x))$
6. $\exists x(P(x, u) \land \neg B(x))$
7. $C(f(j))$

A lógica que lida com esses símbolos é dita **Lógica de Primeira Ordem** ou **Lógica de Predicados**.

4 Estruturas

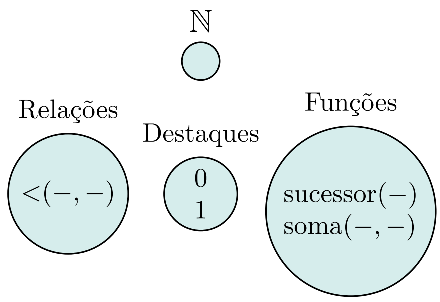
Como vimos, o vocabulário simbólico da lógica de predicados inclui símbolos para representar objetos e predicados, além de símbolos para os conectivos. Assim, a noção de valoração-verdade é incompatível com a lógica de primeira ordem, e necessitamos enriquecê-la para algo que nos permita atribuir valores aos objetos e predicados. Tomamos então, o conceito de **estrutura matemática**.

Estrutura Matemática

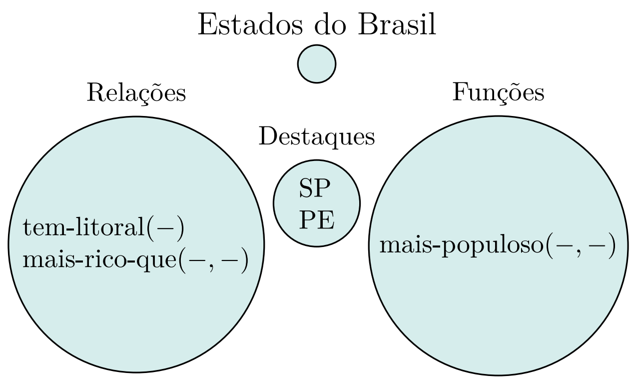
Uma estrutura A é definida por 4 componentes:

- Conjunto de objetos chamado de **domínio** ou **universo** de A – denotado por $dom(A)$.
- Subconjunto de elementos de $dom(A)$ considerados **destaques** ou **constantes**.
- Conjunto de **relações** sobre $dom(A)$, cada uma com sua aridade.
- Conjunto de **funções** sobre $dom(A)$, cada uma com sua aridade.

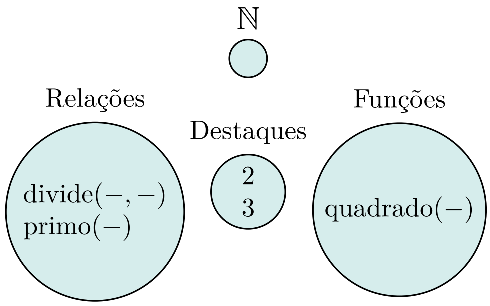
Para ilustrar o conceito, eis três exemplos:



A:



B:



C:

Uma vez definida uma estrutura, podemos criar um vocabulário simbólico sobre a estrutura que nos permita codificar sentenças na lógica de predicados. Tal vocabulário deve dizer quão rica ou simples é a estrutura, envolvendo o número de relações, destaques e funções. Chamamos esse vocabulário de **assinatura** da estrutura.

Assinatura

Seja A uma estrutura. A assinatura L de A é definida pelos seguintes componentes:

- Quantidade de símbolos de destaques e os símbolos.
- Quantidade de símbolos de relações n-árias, onde $n \in \mathbb{N}$, e os símbolos.
- Quantidade de símbolos de funções n-árias, onde $n \in \mathbb{N}$, e os símbolos.

A é dita L-Estrutura.

A assinatura diz respeito somente à quantidade de símbolos. A definição dos mesmos é feita na **linguagem**. Mas, por simplicidade, unimos os dois conceitos. Assim, podemos definir a assinatura de A como:

- 2 símbolos de destaques: a e b;
- 1 símbolo de relação binária: R;
- 1 símbolo de função unária: f;
- 1 símbolo de função binária: g.

Uma vez definidos os símbolos, precisamos dizer o que eles representam em uma estrutura, para que possamos avaliar as sentenças que usam esses símbolos. Tal processo chama-se **interpretação**.

Interpretação

Seja L uma assinatura e A uma L-Estrutura. A interpretação de L em A é uma associação de cada símbolo de L a um elemento de cada componente de A, tal que:

- A cada símbolo c de constante, associa-se um elemento destacado do domínio de A (notação c^A).
- A cada símbolo R de relação de aridade n, associa-se uma relação de A de aridade n (notação R^A).
- A cada símbolo f de relação de aridade n, associa-se uma função de A de aridade n (notação f^A).

Assim, podemos tomar a seguinte interpretação da assinatura de A em A:

- $a^A = 0$ e $b^A = 1$;
- $R^A = <(-,-)$;

- $f^A = \text{sucessor}(-)$ e $g^A = \text{soma}(-,-)$.

Podemos então formalizar sentenças sobre a estrutura A na lógica de predicados:

2 é menor que 3. $R(f(b), f(f(b)))$

Para todo natural x, há um natural y maior que ele. $\forall x \exists y (R(x,y))$

Para todo natural x, a soma entre 1 e x é igual ao sucessor de x. $\forall x (g(b,x) = f(x))$

0 não é sucessor de nenhum natural. $\neg \exists x (f(x) = a)$

Para todo natural x, existem dois naturais cuja soma é x. $\forall x \exists y \exists z (g(y,z) = x)$

4.1 Subestruturas

Como saber se uma estrutura A é subestrutura de uma estrutura B? Se A e B forem simplesmente conjuntos, basta saber se todos os elementos de A também são elementos de B. Mas, considerando os outros componentes das estruturas A e B (relações, destaques e funções), é necessário verificar se esses componentes possuem uma relação entre si que justifique dizer que A está contida em B como estrutura.

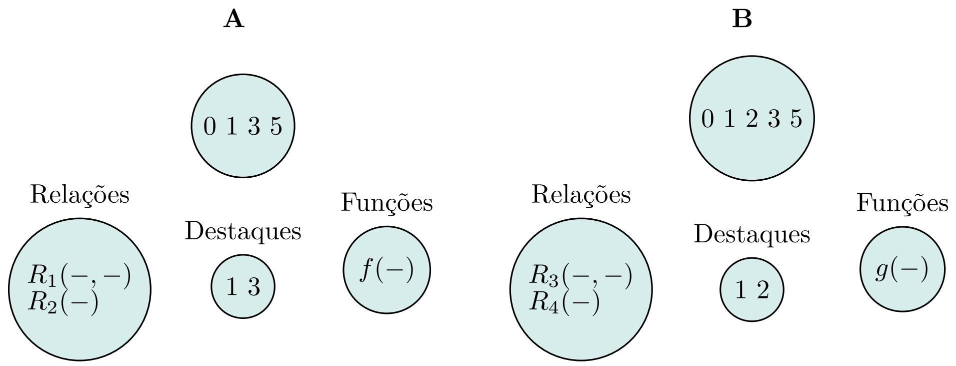
Para definir matematicamente esse possível relacionamento, tomamos emprestado da álgebra a noção de **homomorfismo**: uma função que preserva propriedades.

Homomorfismo

Sejam A e B estruturas de uma mesma assinatura L. Uma função $h : dom(A) \mapsto dom(B)$ é dita homomorfismo de A para B se as condições seguintes forem satisfeitas.

1. Para todo símbolo de constante c de L, $h(c^A) = c^B$;
2. Para todo símbolo de relação n-ária R de L e toda n-upla $(a_1, ..., a_n)$ de elementos de A, $(a_1, ...a_n) \in R^A \to (h(a_1), ..., h(a_n)) \in R^B$;
3. Para todo símbolo de função n-ária f de L e toda n-upla $(a_1, ..., a_n)$ de elementos de A, $h(f^A(a_1, ...a_n)) = f^B(h(a_1), ..., h(a_n))$.

Para ilustrar esse conceito, tomemos duas estruturas A e B:



Suponha que:

$R_1 = \{(0,3),(1,3),(3,5),(5,3)\}$ | $R_3 = \{(0,3),(1,2),(3,5),(2,3),(3,2),(3,3)\}$

$R_2 = \{0,1,5\}$ | $R_4 = \{0,1,2,3,5\}$

$f(0) = 1, f(1) = 1, f(3) = 2, f(5) = 3$ | $g(0) = 0, g(1) = 1, g(2) = 2, g(3) = 3, g(5) = 5$

Seja $h : dom(A) \mapsto dom(B)$ uma função entre as duas estruturas, definida da seguinte forma:

$$\begin{aligned} h(0) &= 1 \\ h(1) &= 1 \\ h(3) &= 2 \\ h(5) &= 3 \end{aligned}$$

h é um homomorfismo de A para B? Vamos verificar cada condição:

1. A 1ª condição diz que os destaques de A são mapeados para destaques de B. Notamos que $h(1) = 1$ e $h(3) = 2$. Assim, a 1ª condição é satisfeita e dizemos que h **preserva destaques**.

2. A 2ª condição diz que se uma tupla de elementos se relaciona em A, então a tupla contendo os mapeamentos desses elementos se relaciona em B. Analisando as relações:

   R_1**:** $(0, 3) \mapsto (h(0), h(3)) = (1, 2) \in R_3$
   $(1, 3) \mapsto (h(1), h(3)) = (1, 2) \in R_3$
   $(3, 5) \mapsto (h(3), h(5)) = (2, 3) \in R_3$
   $(5, 3) \mapsto (h(5), h(3)) = (3, 2) \in R_3$

   R_2**:** $0 \mapsto h(0) = 1 \in R_4$
   $1 \mapsto h(1) = 1 \in R_4$
   $5 \mapsto h(5) = 3 \in R_4$

   Assim, a 2ª condição é satisfeita e dizemos que h **preserva relações**.

3. A 3ª condição diz que mapear a aplicação de uma função em A corresponde a mapear primeiro os argumentos e depois aplicar uma função em B. Analisando as funções:

   $$\begin{aligned} h(f(0)) &= g(h(0)) = g(1) = 1 \\ h(f(1)) &= g(h(1)) = g(1) = 1 \\ h(f(3)) &= g(h(3)) = g(2) = 2 \\ h(f(5)) &= g(h(5)) = g(3) = 3 \end{aligned}$$

   Assim, a 3ª condição é satisfeita e dizemos que h **preserva funções**. Por preservar destaques, relações e funções, h é um homomorfismo de A para B.

4.1.1 Imersão

Um homomorfismo $h : dom(A) \to dom(B)$ é dito **imersão** se:

- h é injetora;
- h satisfaz uma versão mais forte da 2ª condição:
  Para todo símbolo de relação n-ária R de L e toda n-upla $(a_1, ..., a_n)$ de elementos de A, $(a_1, ...a_n) \in R^A \leftrightarrow (h(a_1), ..., h(a_n)) \in R^B$.

A função h do exemplo anterior não é uma imersão, uma vez que, não só ela quebra a primeira condição (pois $h(0) = h(1) = 1$, implicando que h não é injetora) como a segunda ($(5, 5) \notin R_1$, mas $(h(5), h(5)) = (3, 3) \in R_2$). Além da imersão, existem outras variantes do homomorfismo:

- Uma imersão sobrejetora é dita **isomorfismo**.
- Um homomorfismo $h : dom(A) \mapsto dom(A)$ é dito **endomorfismo** de A.
- Um isomorfismo $h : dom(A) \mapsto dom(A)$ é dito **automorfismo** de A.

Agora, podemos remeter ao problema inicial e definir então as condições para que uma estrutura A seja subestrutura de uma estrutura B.

Subestrutura

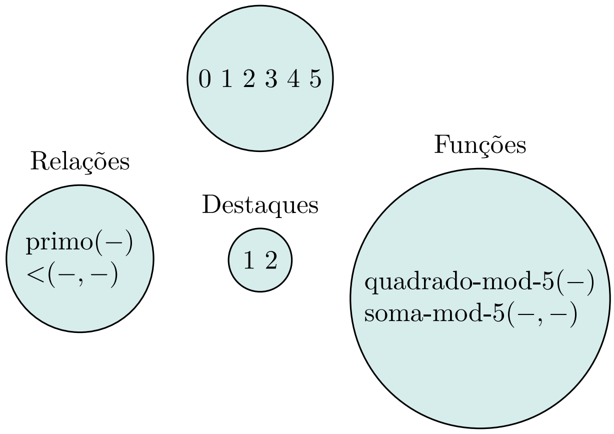
Sejam A e B estruturas de mesma assinatura. Dizemos que A é subestrutura de B se:

1. $dom(A) \subseteq dom(B)$
2. A função identidade $i : dom(A) \mapsto dom(B) \mid i(x) = x$ é uma imersão.

A notação é $A \subseteq B$.

4.1.2 O Problema da Menor Subestrutura

Seja A a estrutura a seguir e $X = \{0, 1, 3\}$ um subconjunto do domínio de A:



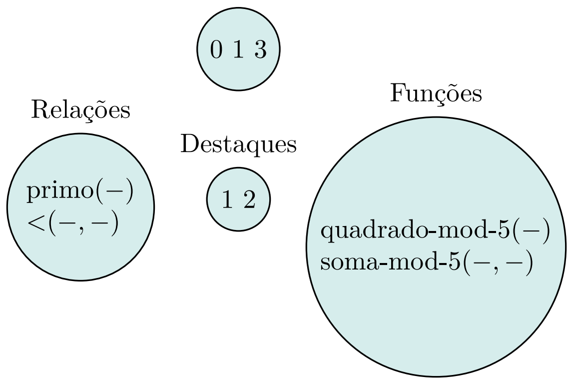
Queremos construir uma subestrutura de A que contenha o menor número de elementos em seu domínio e que contenha X. Estamos diante de um problema de otimização:

Dada: uma L-Estrutura A e um conjunto $X \subseteq dom(A)$;

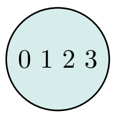
Pergunta-se: qual a menor subestrutura B de A que contém X, ou seja, $B \subseteq A$ e $X \subseteq dom(B)$?

A notação que usamos para B é $\langle X \rangle_A$. Assim, B deve conter os mesmos destaques, relações e funções que A e deve conter X em seu domínio. Além disso, precisamos adicionar elementos ao domínio de B para que a definição de estrutura se mantenha consistente.

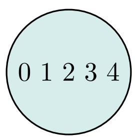
Inicialmente, temos a seguinte estrutura:



Note que ela possui o destaque 2, que não pertence ao domínio. Por definição, o conjunto de destaques é subconjunto do domínio, logo, devemos adicioná-lo a este:



Note também que a função quadrado-mod-5 aplicada a 3 retorna 4, que não é um elemento do domínio. Por definição, o domínio é fechado sob as funções, assim, devemos adicionar 4 ao domínio:



Dessa forma, $\langle X \rangle_A$ é a estrutura com domínio $\{0, 1, 2, 3, 4\}$ e com os mesmos destaques, relações e funções que A.

Podemos sintetizar o procedimento para construir $\langle X \rangle_A$ da seguinte forma:

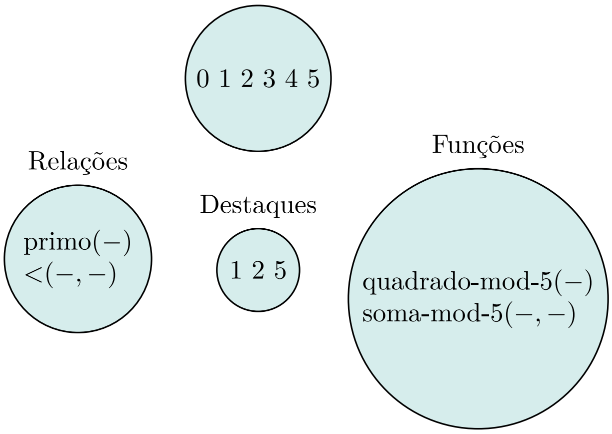
1. Inicialmente, adicione os destaques, funções e relações de A em B e faça $dom(B) = X$.
2. Adicione os destaques de B ao domínio de B.
3. Repita até que nenhum elemento novo seja adicionado:
    3.1 Adicione os conjuntos imagens das funções de B ao domínio de B.

4.1.3 Extensão de uma estrutura

Quando $\langle \emptyset \rangle_A = A$, ou seja, a menor subestrutura de A construída a partir do conjunto vazio como domínio é a própria A, todos os elementos de A são alcançáveis a partir dos destaques e funções de A.

Isso não é verdade na estrutura do exemplo anterior, uma vez que 5 é um elemento inalcançável a partir dos destaques e funções disponíveis e é considerado "sem nome" (é impossível representar 5 por meio de símbolos sobre essa estrutura). Dessa forma, podemos estender a estrutura A, adicionando ao seu conjunto de destaques os elementos inacessíveis de A. A estrutura resultante A' é chamada de **extensão** de A (e A é dita **reduto** de A').

A extensão da estrutura do exemplo anterior é, portanto:



Exercícios Resolvidos

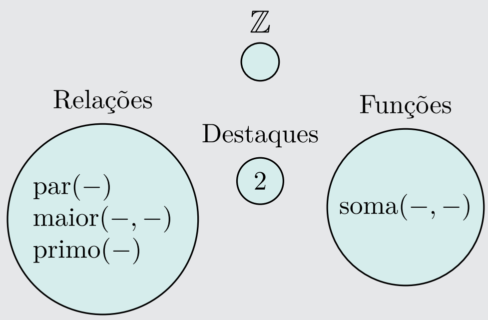
1. Escreva o enunciado abaixo na Lógica de Predicados. Crie uma assinatura e uma estrutura apropriadas para representá-lo, e escolha uma interpretação da assinatura na estrutura.

   Todo inteiro par maior que 2 pode ser escrito como uma soma de dois números primos.
   (**Conjectura de Goldbach**)

   Queremos construir uma estrutura para a qual somos capazes de expressar a Conjectura de Goldbach na lógica de primeira ordem usando sua assinatura. Vamos analisar a sentença:

   - O enunciado fala de inteiros. É razoável admitir, portanto, que o domínio da estrutura é o conjunto dos inteiros.
   - 2 é um inteiro que aparece "destacado" no enunciado. Desse modo, adicionamos-o ao conjunto de destaques.
   - Aparecem as relações: par; maior que; primo.
   - Aparece a função soma.

   Temos, então:



   Podemos definir sua assinatura da seguinte forma:

   - 1 símbolo de destaque: a
   - 2 símbolos de relação unária: R, P
   - 1 símbolo de relação binária: M
   - 1 símbolo de função binária: s

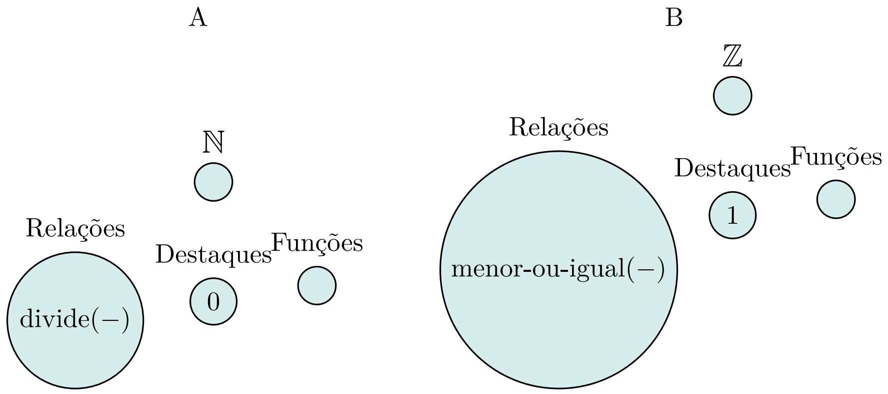
   E então, com base na seguinte interpretação:

   - $a^A = 2$
   - $R^A = \text{Par}(-)$, $P^A = \text{Primo}(-)$, $M^A = \text{Maior}(-,-)$
   - $s^A = \text{soma}(-,-)$

   Podemos escrever a Conjectura de Goldbach na lógica de primeira ordem:

$$\forall x((R(x) \wedge M(x,a)) \to \exists y \exists z(P(y) \wedge P(z) \wedge x = s(y,z)))$$

2. Sejam A e B estruturas de mesma assinatura, definidas a seguir.



(a) Defina um homomorfismo de A para B.

(b) Determine se A é subestrutura de B.

(a) Queremos definir uma função $h : dom(A) \mapsto dom(B)$ e provar que essa função é um homomorfismo de A para B. Temos:

$$h\colon dom(A) \mapsto dom(B)$$
$$h(x) = x + 1$$

- h preserva destaques, pois os destaques de A são mapeados para destaques de B: $h(0) = 1$;
- h preserva relações, pois se um par $(x, y) \in dom(A)$ pertence à relação divide, então $(h(x), h(y)) = (x + 1, y + 1) \in dom(B)$ pertence à relação menor-ou-igual.
- h preserva funções trivialmente, pois não há funções na assinatura das estruturas.

Por preservar destaques, relações e funções, h cumpre as 3 condições de homomorfismo e, portanto, é um.

(b) Queremos verificar se as estruturas satisfazem as duas condições de subestrutura:

1. Como $\mathbb{N} \subseteq \mathbb{Z}$, a primeira condição é satisfeita.
2. Vamos analisar se a função identidade é uma imersão. Ela é injetora, mas não satisfaz a versão mais forte da 2ª condição de homomorfismo. Afinal, o par $(2, 3) \in dom(A)$ não pertence à relação divide, mas o par $(h(2), h(3)) = (2, 3) \in dom(B)$ pertence à relação menor-ou-igual. Mais que isso, a função identidade não é sequer um homomorfismo de A para B, pois não preserva destaques.

Desse modo, A não é subestrutura de B.

Exercícios

1. Para cada sentença abaixo, construa uma assinatura e uma estrutura apropriada para representá-la, e, com base em uma interpretação, expresse-a na lógica de predicados.

   (a) Sejam n, a e b inteiros. Se $n \neq 0$, $n|ab$ e $mdc(n, a) = 1$, então $n|b$. (**Lema de Euclides**)

   (b) Todo grafo planar é 4-colorível. (**Teorema das Quatro Cores**)

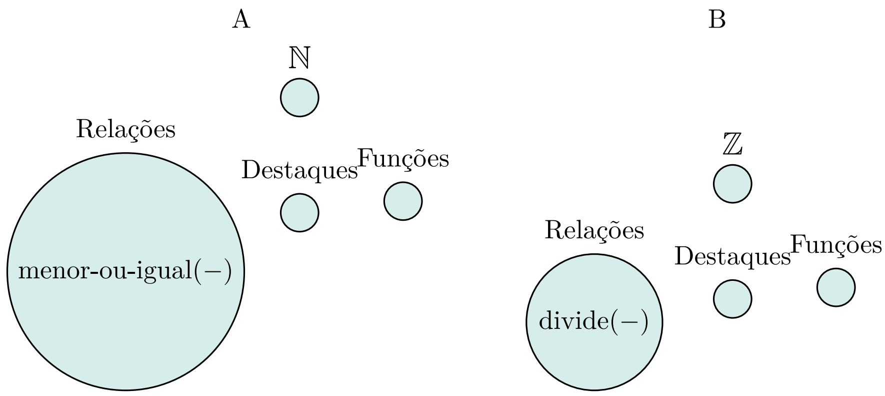
   (c) Para toda proposição, ou ela é verdadeira ou sua negação é verdadeira. (**Lei do Terceiro Excluído**)

   (d) Não existe três inteiros positivos a, b e c tal que $a^n + b^n = c^n$, onde n é um inteiro maior que 2. (**Último Teorema de Fermat**)

   (e) Um inteiro $n > 1$ é primo se, e somente se, $(n-1)! \equiv -1(modn)$. (**Teorema de Wilson**)

   (f) Sejam dois inteiros a e d tal que $d \neq 0$. Então existem dois inteiros q e r tal que $a = qd+r$ e $0 \leq r \leq |d|$. (**Algoritmo da Divisão**)

2. Sejam A e B estruturas definidas abaixo.

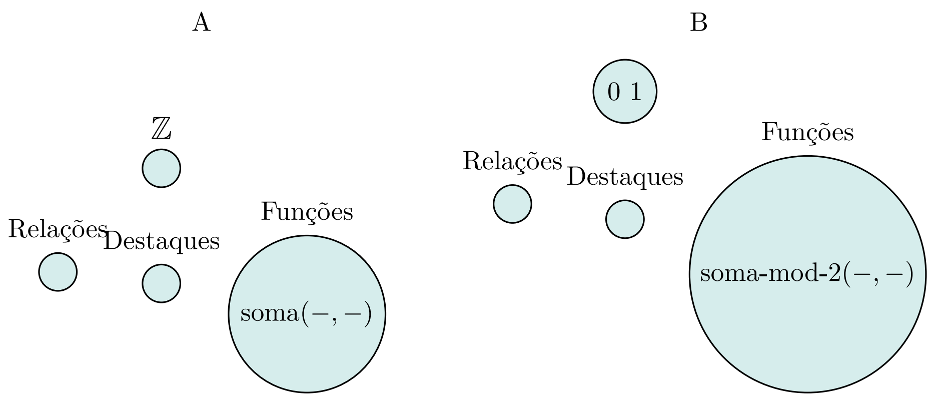


   (a) Defina um homomorfismo de A para B.

   (b) Determine se A é subestrutura de B.

   (c) Defina um automorfismo de B diferente da função identidade.

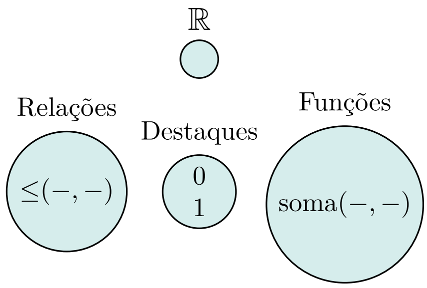
3. Sejam A e B estruturas definidas abaixo.



Mostre que a função h, definida a seguir, é um homomorfismo de A para B.

$$h : dom(A) \mapsto dom(B)$$
$$h(x) = \begin{cases} 0 & \text{se } x \text{ for par} \\ 1 & \text{caso contrário} \end{cases}$$

4. Seja A a estrutura abaixo.



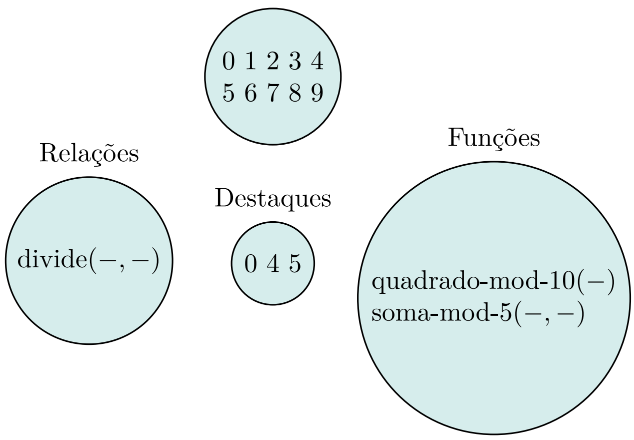
Considere estruturas com os mesmos componentes de A, mas com os domínios abaixo. Determine se cada uma delas pode ser subestrutura de A.

(a) Conjunto dos inteiros positivos

(b) Conjunto dos inteiros

(c) $\{0, 1, 2\}$

5. Dados o conjunto $X = \{2, 6\}$ e a estrutura A abaixo, determine $\langle X \rangle_A$, ou seja, a menor subestrutura de A que contém X no domínio.



5 Sintaxe

Vimos que, além dos conectivos e parênteses, a lógica de predicados envolve mais quatro tipos de símbolos:

Constantes e variáveis São funções de aridade zero e referenciam um objeto do domínio.

Predicados e relações Usamos esses símbolos para denotar alguma propriedade de objetos (predicados) ou uma relação entre objetos (relações). Gottlob Frege, o fundador da lógica de predicados, mostrou como esses conceitos podem ser representados por funções proposicionais (ou seja, funções que retornam verdadeiro ou falso) mesmo que, na prática, sejam conjuntos.

Funções Símbolos que representam funções de referência indireta, ou seja, servem para referenciar um objeto a partir de outros.

Quantificadores Símbolos que denotam quantidade: $\exists$ para denotar "algum objeto" e $\forall$ para denotar "todo objeto".

Assim, o alfabeto Σ da lógica de primeira ordem consiste na união desses quatro tipos de símbolos, acrescidos dos conectivos lógicos e parênteses.

5.1 Fórmulas

Assim como as expressões legítimas da lógica proposicional são chamadas proposições, as expressões legítimas da lógica de primeira ordem são chamadas **fórmulas**.

A unidade básica de uma fórmula é a **fórmula atômica**, que é uma relação entre objetos. Para representar objetos, tomamos o conceito de **termos**.

Termos

Seja L uma linguagem. O conjunto de termos de L é definido indutivamente da seguinte forma:

- Todo símbolo de constante c de L é um termo;
- Toda variável é um termo;
- Se f for um símbolo de função n-ária de L e se $t_1, ..., t_n$ forem termos de L, então $f(t_1, ..., t_n)$ é um termo.

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

Fórmulas Atômicas

Seja L uma linguagem. Uma fórmula atômica é uma palavra sobre o vocabulário simbólico de L com um dos dois formatos:

- $R(t_1, ...t_n)$, onde R é um símbolo de relação n-ária de L e $t_1, ..., t_n$ são termos de L;
- $t_1 = t_2$, onde t_1 e t_2 são termos de L.

Uma fórmula atômica que não contém variáveis é dita **sentença atômica**.

E assim, podemos definir indutivamente o conjunto das expressões legítimas da lógica de predicados, chamado de conjunto das **fórmulas bem formadas** ($FORM$).

Fórmula bem formada

- Toda fórmula atômica é uma fórmula bem formada;
- Se ω é uma fórmula bem formada, então $(\neg\omega)$ é uma fórmula bem formada;
- Se ω_1 e ω_2 são fórmulas bem formadas, então $(\omega_1 \wedge \omega_2)$ é uma fórmula bem formada.
- Se ω_1 e ω_2 são fórmulas bem formadas, então $(\omega_1 \vee \omega_2)$ é uma fórmula bem formada.
- Se ω_1 e ω_2 são fórmulas bem formadas, então $(\omega_1 \to \omega_2)$ é uma fórmula bem formada.
- Se ω é uma fórmula bem formada e x é uma variável livre em ω, então $(\forall x\omega)$ é uma fórmula bem formada;
- Se ω é uma fórmula bem formada e x é uma variável livre em ω, então $(\exists x\omega)$ é uma fórmula bem formada.

Uma fórmula que não contém variáveis livres é dita **sentença**.

Veremos a definição de variável livre a seguir. Temos então que $FORM$ é o fecho indutivo do conjunto base X de fórmulas atômicas sob o conjunto de funções geradoras $F = \{f_\forall, f_\exists, f_\neg, f_\vee, f_\wedge, f_\to\}$, e é possível mostrar que, da forma como definimos, $FORM$ é livremente gerado.

5.2 Variáveis

Uma variável, como vimos, é a representação de um objeto do domínio. Dada uma fórmula $(Qx\omega)$, onde Q é $\forall$ ou $\exists$, dizemos que o **escopo** do quantificador Qx é ω. Por exemplo, na fórmula a seguir:

$$S(x) \vee \exists z(P(z) \wedge \forall x(R(x,y) \to \exists yR(y,x)))$$

O escopo de $\exists z$ é $(P(z) \wedge \forall x(R(x,y) \to \exists yR(y,x))$, o de $\forall x$ é $(R(x,y) \to \exists yR(y,x))$ e o de $\exists y$ é apenas $R(y,x)$. Uma ocorrência de uma variável em uma fórmula é dita **ligada** se, e somente se, a variável está dentro do escopo de um quantificador aplicado a ela ou ela é a ocorrência do quantificador. Uma ocorrência de uma variável em uma fórmula é dita **livre** se, e somente se, essa ocorrência não é ligada.

$$S(x) \vee \exists z(P(z) \wedge \forall x(R(x,y) \to \exists yR(y,x)))$$

Assim, dizemos que uma variável é ligada em uma fórmula se há pelo menos uma ocorrência ligada dela na fórmula e, similarmente, uma variável é livre em uma fórmula se há pelo menos uma ocorrência livre dela na fórmula. Na fórmula anterior, x e y são variáveis livres, enquanto x, y e z são variáveis ligadas (é possível que uma variável seja livre e ligada ao mesmo tempo em uma fórmula, mas em ocorrências diferentes).

Podemos definir uma função recursiva que obtém o conjunto de variáveis livres em uma fórmula:

Conjunto das variáveis livres em uma fórmula

$VL : FORM \mapsto \mathcal{P}(\text{VARIÁVEIS})$
$VL(\varphi) = \{x_1, ..., x_n\}$, se φ é atômica e $x_1, ..., x_n$ ocorrem em φ
$VL((\neg\psi)) = VL(\psi)$
$VL((\rho \wedge \theta)) = VL(\rho) \cup VL(\theta)$
$VL((\rho \vee \theta)) = VL(\rho) \cup VL(\theta)$
$VL((\rho \to \theta)) = VL(\rho) \cup VL(\theta)$
$VL((\forall x\omega)) = VL(\omega) - \{x\}$
$VL((\exists x\omega)) = VL(\omega) - \{x\}$

5.2.1 Substituição de variáveis

Para atribuir um valor t à uma variável livre x em uma fórmula φ (notação $\varphi[t/x]$), devemos substituir todas as ocorrências livres dessa variável na fórmula por esse valor. Assim, queremos definir precisamente esse processo. Temos duas funções: uma aplicada a termos e uma aplicada a fórmulas.

Substituição de uma variável x por um termo t em um termo s

$s[t/x]$: TERM $\times$ TERM $\times$ VARIÁVEIS $\mapsto$ TERM
$x[t/x] = t$
$y[t/x] = y$, se $x \neq y$
$c[t/x] = c$
$f(t_1, ..., t_n)[t/x] = f(t_1[t/x], ..., t_n[t/x])$

Substituição de uma variável x por um termo t em uma fórmula φ

$\varphi[t/x]$: FORM $\times$ TERM $\times$ VARIÁVEIS $\mapsto$ FORM
$R(t_1, ..., t_n)[t/x] = R(t_1[t/x], ..., t_n[t/x])$
$(t_1 = t_2)[t/x] = (t_1[t/x] = t_2[t/x])$
$(\neg\psi)[t/x] = (\neg\psi[t/x])$
$(\rho \wedge \theta)[t/x] = (\rho[t/x] \wedge \theta[t/x])$
$(\rho \vee \theta)[t/x] = (\rho[t/x] \vee \theta[t/x])$
$(\rho \to \theta)[t/x] = (\rho[t/x] \to \theta[t/x])$
$(\forall x\omega)[t/x] = (\forall x\omega)$
$(\forall y\omega)[t/x] = (\forall y\omega[t/x])$, se $x \neq y$
$(\exists x\omega)[t/x] = (\exists x\omega)$
$(\exists y\omega)[t/x] = (\exists y\omega[t/x])$, se $x \neq y$

A função de substituição explora recursivamente a fórmula até encontrar uma ocorrência livre da variável a ser substituída, e então a substitui – isso implica que ocorrências ligadas da variável são ignoradas.

Porém, há uma condição especial para substituição que devemos considerar. Observe a seguinte fórmula φ:

$$\forall x(x = y)$$

Suponha que A seja uma estrutura com mais de um elemento em seu domínio. Isso significa que, para qualquer valor a que atribuirmos a y, a fórmula não diz a verdade nessa estrutura pois nem todo elemento de A é a^A. Isso implica (veremos com mais detalhes no próximo capítulo), que a fórmula φ não é **válida**. Porém, ao substituirmos y pela variável x, temos:

$$\forall x(x = x)$$

Nesse caso, a fórmula sempre diz a verdade, independemente da estrutura analisada, pois qualquer elemento em qualquer estrutura é igual a ele mesmo. Desse modo, a substituição causou uma fórmula não válida se tornar válida, e isso não deve acontecer. Assim, não devemos realizar substituições de um termo t em uma variável x que **causariam ocorrências livres de uma variável em t se tornarem ligadas após a substituição**. Temos então, a noção de **termo livre**.

Termo livre

Dizemos que um termo t está livre para entrar no lugar da variável x em uma fórmula φ se:

- φ é atômica;
- φ é da forma $(\neg\psi)$ e t está livre para entrar no lugar de x em ψ;
- φ é da forma $(\rho * \theta)$, t está livre para entrar no lugar de x em ρ e em θ e $* \in \{\wedge, \vee, \to\}$;
- φ é da forma $(\forall y\omega)$ ou $(\exists y\omega)$, $x = y$ ou $x \neq y$ e $x \notin VL(t)$, e t está livre para entrar no lugar de x em ω.

Assim, executamos essa função antes de proceder com a função de substituição. Na fórmula $\forall x(x = y)$, não podemos realizar a substituição $[x \diagup y]$ pois $x \neq y$, mas $x \in VL(x)$.

Exercícios Resolvidos

1. O termo $f(y,z)$ está livre para entrar no lugar da variável y na fórmula $\forall x(R(y,x) \to \forall z S(z,y))$? E na fórmula $\forall x(R(y,x) \to \forall y S(z,y))$?

   Observe que o enunciado é equivalente a perguntar se as substituições $[f(y,z) / y]$ nas fórmulas $\forall x(R(y,x) \to \forall z S(z,y))$ e $\forall x(R(y,x) \to \forall y S(z,y))$ são possíveis de serem realizadas seguramente.
   Para a primeira fórmula, vemos rapidamente que não, pois a variável z em $f(y,z)$ se tornaria ligada, o que não deve acontecer. Para a segunda fórmula, a única ocorrência da variável y que será substituída é a em $R(y,x)$, pois a em $S(z,y)$ é uma ocorrência ligada. Desse modo, as variáveis de $f(y,z)$ não se tornariam ligadas e, portanto, a substituição é possível. Podemos também responder ambas as questões aplicando a função recursiva.

Exercícios

1. Calcule o conjunto de variáveis livres da fórmula a seguir usando a função recursiva definida no capítulo.
$$\forall x(P(x,y) \rightarrow \exists y(R(z,y) \vee \exists w(S(w,u) \wedge \neg R(w,y))))$$

2. Repita a questão anterior, mas dessa vez calcule o conjunto de variáveis ligadas. Defina uma função recursiva para tal e aplique-a na fórmula.

3. Determine se as substituições a seguir são possíveis de serem realizadas. Se sim, aplique.

   (a) $\forall x(R(y,x) \rightarrow S(y))\ [f(y,z) / y]$

   (b) $\forall z \forall x(R(y,x) \rightarrow S(y))\ [f(y,z) / y]$

   (c) $\forall x(P(x) \vee Q(x,f(y)))\ [g(y) / y]$

   (d) $\forall x(P(x) \vee Q(x,f(y)))\ [g(x) / y]$

   (e) $\forall x(R(x) \wedge S(z))\ [g(z) / y]$

   (f) $\forall x(R(x) \wedge S(z))\ [g(z) / z]$

4. Determine se as afirmações a seguir são verdadeiras ou falsas.

   (a) Se R é um símbolo de relação unária, f é um de função binária e a e b são de constantes, então $R(f(a,b))$ é um termo fechado.

   (b) Se f é um símbolo de função unária e x e y são variáveis, então $f(x) = f(y)$ é uma fórmula atômica.

   (c) Se f é um símbolo de função unária e x e y são variáveis, então $f(x) = f(y)$ é uma sentença atômica.

   (d) Se f é um símbolo de função n-ária e $x_1, ..., x_n$ são variáveis, então $f(x_1, ..., x_n)$ é uma fórmula atômica.

6 Semântica

O matemático Alfred Tarski definiu pioneiramente uma maneira de definir quando uma sentença é verdadeira em uma estrutura, conhecida como a noção de verdade em um "modelo de Tarski" (ou por meio de uma metalinguagem).

Valor-verdade de uma sentença

Seja L uma assinatura, A uma L-Estrutura e $*^A$ uma interpretação dos símbolos de L em A. O valor-verdade de uma sentença φ de L é definida indutivamente da seguinte forma:

- $R(t_1, ..., t_n)^A$ é verdadeira se, e somente se, $(t_1^A, ..., t_n^A) \in R^A$
- $(t_1 = t_2)^A$ é verdadeira se, e somente se, t_1^A for o mesmo elemento que t_2^A
- $(\neg\psi)^A$ é verdadeira se, e somente se, ψ^A for falsa
- $(\rho \wedge \theta)^A$ é verdadeira se, e somente se, ρ^A for verdadeira e θ^A for verdadeira
- $(\rho \vee \theta)^A$ é verdadeira se, e somente se, ρ^A for verdadeira ou θ^A for verdadeira
- $(\rho \to \theta)^A$ é verdadeira se, e somente se, ρ^A for falsa ou θ^A for verdadeira
- $(\forall x\omega)^A$ é verdadeira se, e somente se, ω^A for verdadeira para todo valor de x
- $(\exists x\omega)^A$ é verdadeira se, e somente se, ω^A for verdadeira para algum valor de x

Se φ^A for verdadeira, dizemos que A satisfaz φ sob a interpretação $*^A$.

Satisfatibilidade de uma sentença

Seja φ uma sentença sobre uma assinatura L.

- φ é **satisfatível** se existe uma L-Estrutura A que satisfaz φ sob alguma interpretação de L em A.
- φ é **refutável** se existe uma L-Estrutura A que não satisfaz φ sob alguma interpretação de L em A.
- φ é **válida** se toda L-Estrutura A satisfaz φ sob toda interpretação de L em A.
- φ é **insatisfatível** se toda L-Estrutura A não satisfaz φ sob toda interpretação de L em A.

Seja Γ um conjunto de sentenças sobre L.

- Γ é **satisfatível** se existe uma L-Estrutura A que satisfaz todas as sentenças de Γ sob alguma interpretação de L em A.
- φ é **consequência lógica** de Γ se toda L-Estrutura A que satisfaz Γ também satisfaz φ sob toda interpretação de L em A.

Seja ψ uma sentença sobre uma assinatura L.

- φ e ψ são **logicamente equivalentes** se, para toda L-Estrutura A, A satisfaz ψ se, e somente se, A satisfaz ψ para toda interpretação de L em A.

E para o caso de fórmulas com ocorrências de variáveis livres? Nesse caso, devemos primeiro substituir essas variáveis por termos fechados de uma assinatura.

Satisfatibilidade de uma fórmula

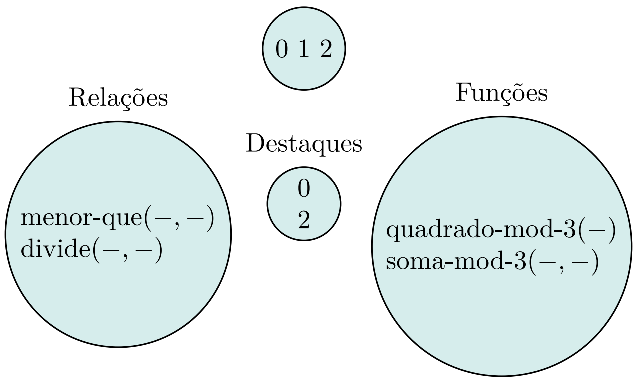
Seja φ uma fórmula sobre uma assinatura L, na qual ocorrem livremente as variáveis $x_1, ..., x_n$.

- φ é **satisfatível** se existe uma L-Estrutura A e termos $a_1, ..., a_n$ de L tal que A satisfaz $\varphi[a_1 / x_1, ..., a_n / x_n]$ sob alguma interpretação de L em A.
- φ é **refutável** se existe uma L-Estrutura A e termos $a_1, ..., a_n$ de L tal que A não satisfaz $\varphi[a_1 / x_1, ..., a_n / x_n]$ sob alguma interpretação de L em A.
- φ é **válida** se, para toda L-Estrutura A e toda n-upla de termos $a_1, ..., a_n$, A satisfaz $\varphi[a_1 / x_1, ..., a_n / x_n]$ sob toda interpretação de L em A.
- φ é **insatisfatível** se, para toda L-Estrutura A e toda n-upla de termos $a_1, ..., a_n$, A não satisfaz $\varphi[a_1 / x_1, ..., a_n / x_n]$ sob toda interpretação de L em A.

6.1 Modelos

Seja L uma assinatura, A uma L-Estrutura e φ uma sentença de L. Dizemos que A é **modelo** para φ se existe uma interpretação de L em A tal que φ^A seja verdadeira. Similarmente, dizemos que A é **contramodelo** para φ se existe uma interpretação de L em A tal que φ^A seja falsa.

Para ilustrá-los, tome a estrutura B a seguir:



A assinatura L de B pode ser definida como:

- 2 símbolos de destaques: a e b;
- 2 símbolo de relação binária: R e S;
- 1 símbolo de função unária: f;
- 1 símbolo de função binária: g.

Ao tomarmos a sentença φ abaixo:

$$\varphi = \exists x \forall y (R(x, y))$$

Notamos que a interpretação R^B = menor-que$(-, -)$ torna φ falsa em B, uma vez que não há um elemento de B que seja menor que 0, 1 e 2. Desse modo, deduzimos que B é contramodelo para φ. Porém, a interpretação R^B = divide$(-, -)$ torna φ verdadeira em B, uma vez que 1 divide 0, 1 e 2. Assim, B também é modelo para φ.

Usando as definições de modelo e contramodelo, podemos então definir os processos de criar um conjunto de sentenças atômicas a partir de uma estrutura de modo a descrevê-la; e de criar uma estrutura a partir de um conjunto de sentenças atômicas de modo a satisfazê-lo.

6.1.1 Diagrama Positivo

Dada uma estrutura A, queremos descrever minuciosamente as componentes dessa estrutura usando sentenças atômicas (uma "descrição lógica" de A). A ideia é escrever sentenças atômicas que descrevem: quais n-uplas do domínio de A pertencem a cada relação n-ária de A; e como funcionam cada função de A. Vamos começar com um exemplo simples, usando a estrutura B definida acima. Tomando a interpretação a seguir como base:

- $a^A = 0$ e $b^A = 2$;
- $R^A =$ menor-que$(-,-)$ e $S^A =$ divide$(-,-)$;
- $f^A =$ quadrado-mod-3$(-)$ e $g^A =$ soma-mod-3$(-,-)$.

Podemos usar essa linguagem simbólica para descrever B: $R(a,b)$, $R(a,f(b))$, $R(f(b),b)$, $R(g(a,a),b)$, $R(g(a,a),f(b))$, $S(f(b),a)$, $S(f(b),b)$, $S(b,a)$, $S(f(b),f(b))$, $S(b,b)$, $a = g(a,a)$, $b = g(f(b),f(b))$, $f(b) = g(b,b)$, $a = f(a)$, $b = g(b,a)$...

O que fizemos foi descrever uma base para que possamos obter o conjunto de todas as sentenças atômicas que são verdadeiras em B sob a interpretação dada. Chamamos esse conjunto de **diagrama positivo** de B. Para alcançar esse conjunto, precisamos completar essa base com o que falta para que ele inclua todas as sentenças atômicas verdadeiras em B. Assim, por exemplo, se $R(a,b)$ está nesse conjunto e $a = f(a)$ está nesse conjunto, então devemos incluir $R(f(a),b)$. Esse processo de completação é dito **fecho sob igualdade**.

Fecho sob igualdade

Seja T um conjunto de sentenças atômicas sobre uma linguagem L. O fecho sob igualdade de T é o menor conjunto T' de sentenças atômicas sobre L tal que:

- T' contém T;
- Se T' contém $\varphi(t)$ e contém $(t = t')$, então T' contém $\varphi(t')$.
- T' contém $(t = t)$ para todo termo fechado t de L.

Diagrama positivo

Seja L uma assinatura e A uma L-Estrutura. O conjunto de todas as sentenças atômicas sobre L que são verdadeiras em A sob alguma interpretação de L em A, ou seja, as sentenças atômicas para as quais A é modelo, é dito diagrama positivo de A (notação $diag^+(A)$).
Caso $\langle\emptyset\rangle_A \neq A$, usamos a extensão de A para gerar o diagrama positivo.

Construção do diagrama positivo

O processo de construção do diagrama positivo de uma estrutura A se dá da seguinte forma:

1. Inclua todos os elementos inalcançáveis do domínio de A como destaques em A.
2. Para todo símbolo de relação n-ária de L e termos $t_1, ..., t_n$, inclua $R(t_1, ..., t_n)$ se $t_1^A, ..., t_n^A \in R^A$.
3. Para todo símbolo de função n-ária de L e termos $t_1, ..., t_n, t_{n+1}$, inclua $f(t_1, ..., t_n) = t_{n+1}$ se $f^A(t_1^A, ..., t_n^A)$ for o mesmo elemento que t_{n+1}^A.

6.1.2 Modelo Canônico

Suponha que tenhamos o seguinte conjunto T de sentenças atômicas:

$$T = \{R(a), S(g(b), b), g(b) = g(a), g(b) = a, f(g(a), a) = b, S(a, a), R(g(b))\}$$

De modo inverso ao diagrama positivo, queremos construir uma estrutura D a partir de T, tal que D seja modelo para todas as sentenças de T. Para tal, precisamos definir as quatro componentes de D: conjunto domínio, conjunto de destaques, conjunto de relações e conjunto de funções.

Domínio O domínio, a princípio, seria o conjunto de todos os termos fechados que aparecem em T. Porém, uma vez que D deve satisfazer $g(b) = a$ e $g(b) = g(a)$, nessa estrutura, a, $g(b)$ e $g(a)$ devem ser o mesmo elemento. Desse modo, não faz sentido colocarmos os três no domínio, pois conjuntos não admitem repetição de elementos. Iremos então agrupar esses termos iguais em uma classe e colocar no domínio apenas algum termo que os represente, por exemplo, $a^{\sim}$. Assim, repetindo essa análise nas outras igualdades, o conjunto universo de D é

$$a^{\sim} \ b^{\sim}$$

Destaques O conjunto de destaques é subconjunto do domínio, e deve conter todos os destaques que ocorrem em T:

$$a^{\sim} \ b^{\sim}$$

Relações O conjunto de relações contém todas as relações que ocorrem em T. Porém, uma vez que D deve satisfazer as sentenças de T, é necessário incluir tais elementos nas relações.

$$R(-)$$
$$S(-,-)$$

Onde $R^D = \{a^{\sim}\}$ e $S^D = \{(a^{\sim}, b^{\sim}), (a^{\sim}, a^{\sim})\}$.

Funções O conjunto de funções contém todas as funções que ocorrem em T. Porém, as funções de D possuem como domínio o universo de D e devemos, portanto, definí-las.

$$f(-,-)$$
$$g(-)$$

Onde $f^D(a^{\sim}, a^{\sim}) = b^{\sim}$, $g^D(a^{\sim}) = a^{\sim}$ e $g^D(b^{\sim}) = a^{\sim}$.

D é uma estrutura que é modelo para todas as sentenças de T e é chamada **modelo canônico** de T. Antes de definir formalmente esse tipo de estrutura, precisamos definir o processo de selecionar um representante de uma classe que definimos na construção do domínio. Usaremos a noção de **classes de equivalência**.

A relação $\sim$

Seja L uma assinatura e T um conjunto de sentenças atômicas que é o diagrama positivo de alguma L-Estrutura. Podemos definir a relação binária $\sim$ da seguinte forma:

$$\sim = \{(t_1, t_2) \mid (t_1 = t_2) \in T\}$$

$\sim$ é reflexiva: Para todo $t \in L$, $(t = t) \in T$, logo $(t, t) \in \sim$.

$\sim$ é simétrica: Se $(t_1, t_2) \in \sim$, significa que $(t_1 = t_2) \in T$. Como a igualdade é comutativa, $(t_2 = t_1) \in T$ e, portanto, $(t_2, t_1) \in \sim$.

$\sim$ é transitiva: Se $(t_1, t_2) \in \sim$ e $(t_2, t_3) \in \sim$, significa que $(t_1 = t_2) \in T$ e $(t_2 = t_3) \in T$. Como a igualdade é transitiva, $(t_1 = t_3) \in T$. Desse modo, $(t_1, t_3) \in \sim$.

Sendo reflexiva, simétrica e transitiva, $\sim$ é uma **relação de equivalência** e, portanto, particiona o conjunto domínio de modo a criar classes de equivalência.

Modelo canônico

Seja T um conjunto de sentenças atômicas de uma linguagem L. A L-Estrutura B mais genérica possível que é modelo para todas as sentenças de T é dita modelo canônico de T e é definida da seguinte forma:

Domínio O domínio é o conjunto dos representantes das classes de equivalência $t^{\sim}$ dos termos fechados de L no conjunto T.

Destaques O conjunto de destaques contém todos os representantes das classes de equivalência $c^{\sim}$, onde c é um símbolo de constante de L.

Relações Seja R um símbolo de relação n-ária de L e $t_1, ..., t_n$ termos de L. Então $(t_1^{\sim}, ..., t_n^{\sim}) \in R^B$ se, e somente se, $R(t_1, ..., t_n) \in T$.

Funções Seja f um símbolo de relação n-ária de L e $t_1, ..., t_n$ termos de L. Então $f^B(t_1^{\sim}, ..., t_n^{\sim}) = f(t_1, ..., t_n)^{\sim}$.

O modelo canônico D de um conjunto de sentenças T é parecida com qualquer que tenha sido a estrutura B que foi o seu modelo "original" (ou seja, T é o diagrama positivo de B). D é chamada de **canônico** pois serve de referencial para todos os modelos de T. Isso significa que podemos construir um homomorfismo de D para qualquer outro modelo B de T:

$$h : dom(D) \mapsto dom(B)$$
$$h(t^{\sim}) = t^B$$

1. Para todo símbolo de constante c de L, $h(c^D) = c^B$. Como $c^D = c^{\sim}$, h preserva destaques.

2. Para todo símbolo de relação n-ária R de L e toda n-upla $t_1, ..., t_n$ de L, se $(t_1^{\sim}, ...t_n^{\sim}) \in R^D$, então $R(t_1, ..., t_n) \in T$. Dessa forma, $(t_1^B, ..., t_n^B) \in R^B$ e, portanto, $(h(t_1^{\sim}), ..., h(t_n^{\sim}))$. Assim, h preserva relações.

3. Para todo símbolo de função n-ária f de L e toda n-upla $t_1, ..., t_n$ de L, por definição, $f^D(t_1^{\sim}, ..., t_n^{\sim}) = f(t_1, ..., t_n)^{\sim}$. Dessa forma, $h(f^D(t_1^{\sim}, ..., t_n^{\sim})) = h(f(t_1, ..., t_n)^{\sim}) = f(t_1, ..., t_n)^D$ $= f^D(t_1^D, ..., t_n^D) = f^D(h(t_1^{\sim}), ..., h(t_n^{\sim}))$. Assim, h preserva funções.

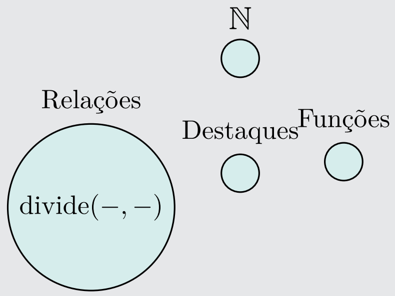
Por preservar destaques, relações e funções, h é um homomorfismo.

Finalmente, uma vez que podemos construir um modelo para qualquer conjunto de sentenças atômicas, **todo conjunto de sentenças atômicas é satisfatível**.

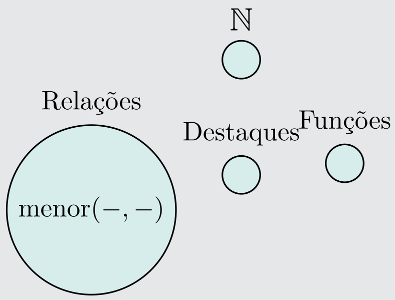
Exercícios Resolvidos

1. Seja $\sigma = \exists x \forall y (R(x,y) \vee R(y,x))$. Defina duas estruturas A e B tal que A seja modelo de σ e B seja contramodelo de σ.

Queremos construir uma estrutura A que satisfaça σ sob alguma interpretação. Temos:

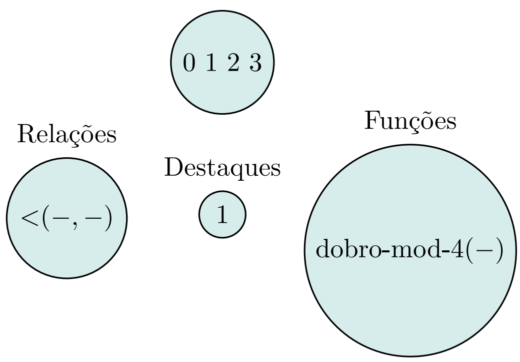


A é modelo para σ pois, ao tomarmos R como divide$(-,-)$, temos que 1 é um natural que divide todos os outros, tornando a fórmula $R(x,y)$ verdadeira para qualquer valor de y, e, por consequência, a disjunção.
Agora, queremos construir uma estrutura B que refute σ sob alguma interpretação. Isso equivale a dizer que B satisfaz $\neg\sigma \equiv \forall x \exists y (\neg R(x,y) \wedge \neg R(y,x))$ sob alguma interpretação. Temos:



B é modelo para $\neg\sigma$ (e, portanto, contramodelo para σ) pois, ao tomarmos R como menor$(-,-)$, para todo natural x, podemos tomar y como x. Desse modo, nem x é menor que y e nem y é menor que x, satisfazendo a conjunção.

2. Esboce o diagrama positivo da estrutura A abaixo.



Queremos construir um esboço do conjunto que contém todas as sentenças atômicas que são verdadeiras em A sob alguma interpretação. Primeiramente, notemos que $\langle \emptyset \rangle_A \neq A$, já que 3 não é alcançável pelos destaques e funções de A. Desse modo, adicionamos 3 aos destaques, assim como mais um símbolo de destaque na assinatura:

- 2 símbolos de destaque: a, b
- 1 símbolo de relação binária: M
- 1 símbolo de função unária: d

E então, escolhemos uma interpretação:

- $a^A = 1$; $b^A = 3$
- $M^A = <(-,-)$
- $d^A =$ dobro-mod-4$(-)$

Assim, temos:
$diag^+(A) = \{M(a,b), M(a,d(a)), M(d(a),b), M(d(d(a)),a), M(d(d(a)),d(a)), M(d(d(a)),b), d(a) = d(b), d(d(a)) = d(d(d(a))), ...\}$

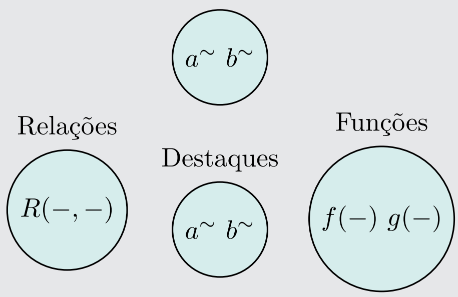
3. Construa o modelo canônico do conjunto de sentenças atômicas T a seguir.

$$T = \{R(a,b), f(a) = g(b), a = f(a), b = f(b), g(a) = f(a)\}$$

Queremos construir a estrutura mais genérica possível que seja modelo para todas as quatro sentenças de T. Primeiramente, devemos identificar as classes de equivalência da relação $\sim$:

- Notemos que $f(a) = g(b)$, $f(a) = a$ e $g(a) = f(a)$. Desse modo, a, $f(a)$, $g(b)$ e $g(a)$ constituem uma classe de equivalência, e tomaremos $a^\sim$ como sua representante.
- Como $b = f(b)$, b e $f(b)$ constituem outra classe de equivalência, e tomaremos $b^\sim$ como sua representante.

Assim, temos a estrutura D:



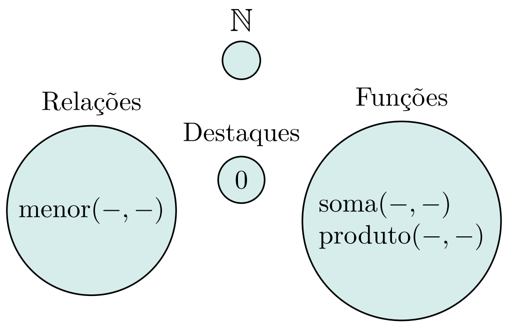
Onde:

$R^D = \{(a^\sim, b^\sim)\}$

$f^D(a^\sim) = a^\sim$, $f^D(b^\sim) = b^\sim$, $g^D(a^\sim) = a^\sim$ e $g^D(b^\sim) = a^\sim$

Exercícios

1. Seja A a estrutura abaixo:



   Determine se A é modelo ou contramodelo (ou ambos) para as sentenças a seguir. Considere que a, R e f são símbolos da assinatura de A.

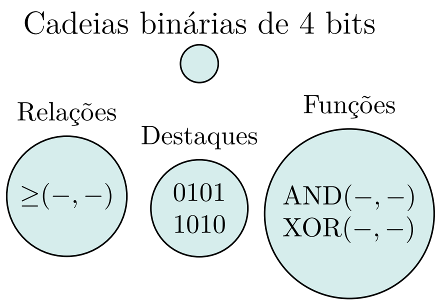
   (a) $\forall x \exists y (f(x, y) = a)$

   (b) $\forall x \forall y (f(x, y) = f(y, x))$

   (c) $\forall x \forall y (R(x, y) \to \exists p (\neg(p = a) \land f(x, p) = y))$

   (d) $\forall x \forall y (\exists z (f(x, z) = y) \to R(x, y))$

2. Seja B a estrutura abaixo:



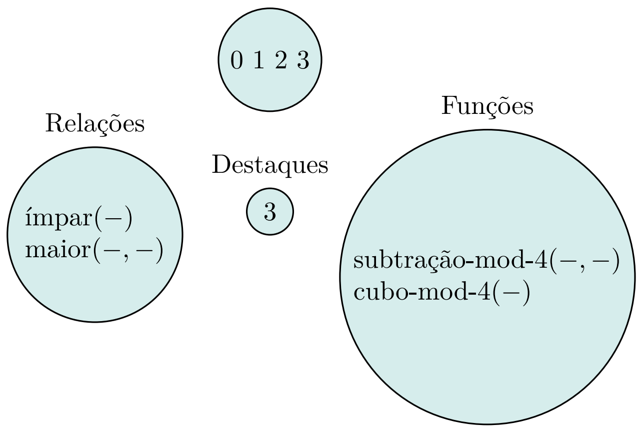
   Defina uma assinatura L para A, e então escreva uma sentença sobre L para a qual A é tanto modelo quanto contramodelo.

3. Construa o modelo canônico dos seguintes conjuntos de sentenças atômicas:

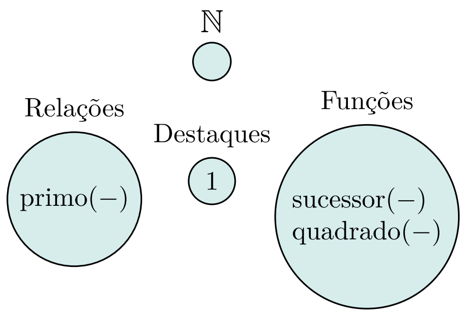
   (a) $\{f(a) = h(b, c), f(a) = f(c), b = g(a, b), R(a), R(c), S(b, c)\}$

   (b) $\{R(a), M(f(b), b), f(a) = f(b), f(b) = a, h(f(a), a) = b, h(a, b) = f(f(a)), h(b, a) = h(h(f(a), a), b), R(h(f(h(a, b)), b)), M(a, a)\}$

4. Para cada estrutura a seguir, defina uma assinatura e uma interpretação da assinatura na estrutura, e construa um esboço do seu diagrama positivo.


0 1 2 3
Relações
ímpar(−)
maior(−, −)
Destaques
3
Funções
subtração-mod-4(−, −)
cubo-mod-4(−)

(a)


ℕ
Relações
primo(−)
Destaques
1
Funções
sucessor(−)
quadrado(−)

(b)

7 Problema da Satisfatibilidade

Dada: uma fórmula φ;

Pergunta-se: φ é satisfatível?

Retomamos o Problema da Satisfatibilidade (SAT), dessa vez sob a perspectiva da lógica de primeira ordem. Porém, temos uma peculiaridade envolvendo satisfatibilidade aqui. Em lógica proposicional, podemos avaliar se uma proposição é tautologia verificando se ela é verdadeira sob todas as valorações-verdade possíveis para a proposição. Essa maneira de força-bruta de resolver não funciona em lógica de predicados, uma vez que precisaríamos avaliar uma fórmula sob todas as interpretações em todas as estruturas (de mesma assinatura da fórmula), e existe um número infinito de estruturas. Assim, para determinar invalidez de uma fórmula, os algoritmos devem ser capazes de detectar que ela é falsa em um número finito de testes. Mas há casos em que isso simplesmente não é possível devido à natureza da fórmula. Desse modo, os algoritmos de primeira ordem conseguem decidir se uma fórmula é válida se ela de fato for válida, mas, caso não seja, eles podem nunca terminar. Alonzo Church e Alan Turing provaram em 1936 que um algoritmo que consegue decidir corretamente se qualquer fórmula é válida (ou insatisfatível) ou não não pode existir. Desse modo, classificamos a lógica de primeira ordem como **semi-decidível**.

7.1 Método da Resolução

A filosofia do método persiste: ser eficiente para certas entradas e eficiente em reconhecer essas entradas. Os conceitos como a regra da resolução e a forma normal conjuntiva continuam valendo, mas dessa vez, temos diferenças cruciais.

7.1.1 O Problema da Unificação de Termos

Se em uma cláusula C_1 há um literal L e em outra cláusula C_2 há o literal $\neg L$, a regra da resolução determina que podemos criar uma cláusula nova – o resolvente de C_1 e C_2, sendo a disjunção de todos os outros literais de C_1 e C_2. Porém, uma condição importante é que L em C_1 seja idêntico a L (de $\neg L$) em C_2.

Em lógica proposicional, esse problema é trivial pois avaliamos apenas variáveis. Mas em primeira ordem, caso dois literais não sejam idênticos, devemos ser capazes de detectar se eles **podem** ser idênticos, utilizando de substituição de variáveis. Por exemplo, podemos aplicar a regra em $R(a)$ e $\neg R(x)$ se tomarmos $[a \diagup x]$, mas o mesmo não é possível com $P(a)$ e $\neg P(b)$, pois não há substituição que torne os literais idênticos. De modo geral, precisamos de um procedimento que determine se dois literais podem ser idênticos.

Dados: dois termos t_1 e t_2;

Pergunta-se: existe uma substituição de variáveis que torne t_1 e t_2 idênticos?

Caso tal substituição exista, dizemos que t_1 e t_2 são termos **unificáveis** e a substituição é dita **unificadora**. Em 1930, o matemático Jacques Herbrand definiu um algoritmo simples que resolve o problema usando regras de transformação de um conjunto de equações. Uma **equação** é um par de termos $(s = t)$. Um **sistema de equações** S é um multiconjunto de equações e uma substituição Θ é unificadora de S se ela unifica todas as equações de S. O conjunto de todas as substituições unificadoras de S é denotado por $U(S)$. Esse conjunto pode conter várias substituições possíveis, mas uma delas é a "ótima". Por exemplo, para os termos abaixo:

$$f(y)) \quad \text{e} \quad f(g(z))$$

A substituição $\Theta_1 = [g(b)/y, b/z]$ unifica os termos, mas não é a única possível. $\Theta_2 = [g(z)/y]$ também os unifica e é mais simples. A substituição mais simples que unifica dois termos dentre as possíveis é dita **unificadora mais geral**.

O algoritmo de Herbrand usa como base o conceito de **forma resolvida**: uma equação está na forma resolvida em um sistema S se for da forma $v = t$ (variável = termo) e v for uma variável resolvida, ou seja, v não ocorre em t e em nenhuma outra equação de S. Caso todas as equações de S estejam na forma resolvida, o sistema é unificável e o algoritmo devolve a substituição unificadora mais geral de S.

Regras de Transformação

Eliminação de Equações Triviais $S \cup \{t \stackrel{?}{=} t\} \Rightarrow S$

Decomposição de Termos $S \cup \{f(t_1, ..., t_n) \stackrel{?}{=} f(s_1, ..., s_n)\} \Rightarrow S \cup \{t_1 \stackrel{?}{=} s_1, ..., t_n \stackrel{?}{=} s_n\}$

- $S \cup \{f(t_1, ..., t_n) \stackrel{?}{=} g(t_1, ..., t_m)\}$, onde $f \neq g$ ou $n \neq m \Rightarrow$ *falha*.

Eliminação de Variáveis $S \cup \{x \stackrel{?}{=} t\} \Rightarrow S[x/t] \cup \{x \stackrel{?}{=} t\}$, se x não ocorre em t.

- Se x ocorre em $t \Rightarrow$ *falha*.

Algoritmo de Herbrand

Entrada: um sistema de equações S;

Saída: Se S for unificável, a unificadora mais geral de S; caso contrário, S não é unificável.

1. Para cada equação s de S:
   1.1 Se s estiver na forma resolvida, passe para a próxima.
   1.2 Caso contrário, aplique uma das regras de transformação, com prioridade decrescente de cima pra baixo, e vá para o início.
   1.3 Se não for possível aplicar uma regra, retorne **não é unificável**.
2. Caso todas as equações de s estejam na forma resolvida, retorne a **unificadora mais geral** formada pelas equações presentes em S.

Para ilustrar, vamos executar o algoritmo sobre o seguinte conjunto de equações S:

$$S = \{f(g(z), x) = f(y, x), f(y, x) = f(y, h(a)), f(g(z), x) = f(y, h(a))\}$$

$$\begin{array}{rl}
\textbf{DT} & \{g(z) \stackrel{?}{=} y, x \stackrel{?}{=} x, f(y, x) \stackrel{?}{=} f(y, h(a)), f(g(z), x) \stackrel{?}{=} f(y, h(a))\} \\
\textbf{EV} & \{\mathbf{y} \stackrel{?}{=} \mathbf{g(z)}, x \stackrel{?}{=} x, f(g(z), x) \stackrel{?}{=} f(g(z), h(a)), f(g(z), x) \stackrel{?}{=} f(g(z), h(a))\} \\
\textbf{EET} & \{\mathbf{y} \stackrel{?}{=} \mathbf{g(z)}, f(g(z), x) \stackrel{?}{=} f(g(z), h(a)), f(g(z), x) \stackrel{?}{=} f(g(z), h(a))\} \\
\textbf{DT} & \{\mathbf{y} \stackrel{?}{=} \mathbf{g(z)}, g(z) \stackrel{?}{=} g(z), x \stackrel{?}{=} h(a), f(g(z), x) \stackrel{?}{=} f(g(z), h(a))\} \\
\textbf{EET} & \{\mathbf{y} \stackrel{?}{=} \mathbf{g(z)}, x \stackrel{?}{=} h(a), f(g(z), x) \stackrel{?}{=} f(g(z), h(a))\} \\
\textbf{EV} & \{\mathbf{y} \stackrel{?}{=} \mathbf{g(z)}, \mathbf{x} \stackrel{?}{=} \mathbf{h(a)}, f(g(z), h(a)) \stackrel{?}{=} f(g(z), h(a))\} \\
\textbf{DT} & \{\mathbf{y} \stackrel{?}{=} \mathbf{g(z)}, \mathbf{x} \stackrel{?}{=} \mathbf{h(a)}, g(z) \stackrel{?}{=} g(z), h(a) \stackrel{?}{=} h(a)\} \\
\textbf{EET} & \{\mathbf{y} \stackrel{?}{=} \mathbf{g(z)}, \mathbf{x} \stackrel{?}{=} \mathbf{h(a)}, h(a) \stackrel{?}{=} h(a)\} \\
\textbf{EET} & \{\mathbf{y} \stackrel{?}{=} \mathbf{g(z)}, \mathbf{x} \stackrel{?}{=} \mathbf{h(a)}\}
\end{array}$$

S é unificável e a unificadora mais geral de S é $\Theta = [g(z)/y, h(a)/x]$. Isso significa que, para os termos serem idênticos, basta substituir x por $h(a)$ e y por $g(z)$. Vejamos agora um exemplo onde o algoritmo falha:

$$S' = \{g(f(x, x)) = g(f(h(a), g(b)))\}$$

$$\begin{array}{ll}
\mathbf{DT} & \{f(x,x) \stackrel{?}{=} f(h(a), g(b))\} \\
\mathbf{DT} & \{x \stackrel{?}{=} h(a), x \stackrel{?}{=} g(b)\} \\
\mathbf{EV} & \{\mathbf{x} \stackrel{?}{=} \mathbf{h}(\mathbf{a}), h(a) \stackrel{?}{=} g(b)\} \\
 & \{\mathbf{x} \stackrel{?}{=} \mathbf{h}(\mathbf{a}), h(a) \stackrel{?}{=} g(b)\}
\end{array}$$

Uma vez que $h \neq g$, o algoritmo falha ao tentar aplicar uma regra em $h(a) \stackrel{?}{=} g(b)$. Desse modo, o sistema não é unificável. De fato: não existe nenhuma substituição de variáveis que torne $x = h(a)$ e $x = g(b)$ ao mesmo tempo.

No contexto de resolução, portanto, usaremos o algoritmo de Herbrand sempre que tentarmos aplicar a regra da resolução em dois literais complementares. Caso sejam unificáveis, procedemos com o resolvente e **aplicamos a unificadora mais geral neste**. Caso contrário, não aplicaremos a regra.

7.1.2 Remoção dos quantificadores

Já concluímos um dos problemas referentes ao método da resolução, que se preocupava com a identicidade de literais. Agora, veremos o outro grande problema, sobre a sintaxe da entrada. As entradas do método da resolução devem estar na forma normal conjuntiva, que, como sabemos, é uma conjunção de cláusulas. Porém, as fórmulas da lógica de primeira ordem podem possuir quantificadores, que não estão presentes nessa definição. Dessa maneira, precisamos removê-los. A estratégia que usaremos será a seguinte: colocar todos os quantificadores no início da fórmula, e então removê-los.

Forma Normal Prenex

Uma fórmula φ está na **forma normal prenex** (*prefixed normal expression*) se, e somente se, ela tem o seguinte formato:

$$(Q_1 x_1 Q_2 x_2 ... Q_n x_n)(\psi)$$

Onde $(Q_1 x_1 Q_2 x_2 ... Q_n x_n)$ é uma parte com quantificadores e chamada de **prefixo** de φ e ψ é uma fórmula sem quantificadores chamada de **matriz** de φ.

Teorema 7.1.1

Para toda fórmula φ, existe uma fórmula ψ tal que:

- ψ está na forma normal prenex;
- ψ é logicamente equivalente a φ.

Prova (transformação para a FNP)

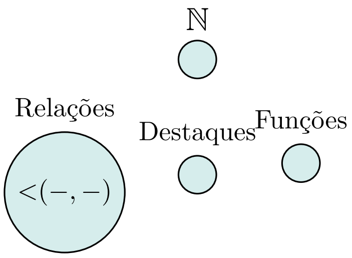
Para provar o teorema, faremos uma prova por construção mostrando um método que transforma qualquer fórmula φ para outra ψ logicamente equivalente e na forma normal prenex.

1. Elimine os conectivos $\rightarrow$ e $\leftrightarrow$.
2. Aplique repetidamente, além das equivalências conhecidas $(\neg\neg L) \equiv L$ (lei da dupla negação), as leis de De Morgan e a propriedade distributiva, as equivalências a seguir, até que a fórmula esteja na forma normal prenex.
   - $\neg\forall x(\omega) \equiv \exists x(\neg\omega)$
   - $\neg\exists x(\omega) \equiv \forall x(\neg\omega)$
   - $\forall x(\varphi(x)) \wedge \forall x(\psi(x)) \equiv \forall x(\varphi(x) \wedge \psi(x))$
   - $\exists x(\varphi(x)) \vee \exists x(\psi(x)) \equiv \exists x(\varphi(x) \vee \psi(x))$
   - $Qx(\varphi(x)) \equiv Qy(\varphi(y))$, onde $Q \in \{\forall, \exists\}$
   - $Qx(\varphi(x)) \wedge \psi \equiv Qx(\varphi(x) \wedge \psi)$, onde $Q \in \{\forall, \exists\}$
   - $Qx(\varphi(x)) \vee \psi \equiv Qx(\varphi(x) \vee \psi)$, onde $Q \in \{\forall, \exists\}$

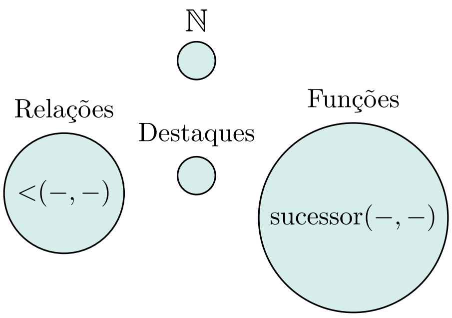
Forma Padrão de Skolem

Uma fórmula φ na forma normal prenex está na **forma padrão de Skolem** se, e somente se, o prefixo de φ não contém quantificadores existenciais. Em 1920, o lógico-matemático Thoralf Skolem definiu um método para remoção de quantificadores existenciais, que acabou ganhando seu nome: o **método da skolemização**.

Suponha a seguinte estrutura A:



A sentença $\varphi = \forall x \exists y R(x, y)$ é verdadeira em A, pois para todo natural x existe um outro maior que ele, por exemplo, seu sucessor. Dessa forma, podemos retirar o existencial, substituindo a ocorrência de sua variável pela função sucessor aplicada a x, ou seja, $\varphi' = \forall x R(x, f(x))$.



φ' é verdadeira nessa nova estrutura que é idêntica a A, adicionando-se a função f. A skolemização é, portanto, o processo de remover quantificadores existenciais, substituindo as ocorrências das variáveis desses quantificadores por funções aplicadas a variáveis universalmente quantificadas anteriores a esses existenciais.

Teorema 7.1.2: Teorema de Löwenheim-Skolem

Seja φ uma fórmula na forma normal prenex sobre uma assinatura L. Seja ψ a fórmula resultante da remoção de cada quantificador existencial que ocorre em φ e cujas variáveis correspondentes são substituídas por um termo do tipo $f(x_1, ..., x_n)$, onde f é um novo símbolo de função e $x_1, ..., x_n$ são variáveis universalmente quantificadas imediatamente anteriores a esse existencial. Então, se existe uma L-Estrutura A que é modelo para φ, é possível construir uma L'-Estrutura B que é modelo para ψ, acrescentando a A uma interpretação para cada símbolo novo de função em L'.

Caso não haja quantificadores universais anteriores a um existencial, a variável correspondente deste é substituída por um novo símbolo de constante (função de aridade zero). Uma função nova adicionada na assinatura é chamada **função de Skolem**, e uma constante nova é chamada **constante de Skolem**. A skolemização não preserva a equivalência das fórmulas resultante e original, mas preserva a satisfatibilidade: a original é satisfatível se, e somente se, a resultante é satisfatível.

Para ilustrar o processo de remoção de quantificadores, vejamos as seguintes fórmulas:

- $\forall x P(x) \rightarrow \exists x S(x)$:
  $\equiv \neg \forall x P(x) \lor \exists x S(x)$
  $\equiv \exists x(\neg P(x)) \lor \exists x S(x)$
  $\equiv \underline{\exists x}(\neg P(x) \lor S(x))$ **FNP**

$\exists x \Rightarrow$ adicionando uma constante de Skolem a na assinatura;
$\equiv \neg P(a) \vee S(a)$ **FPS**

- $\exists x(R(x,y) \rightarrow \forall y(P(z,y) \wedge \exists w(S(w,u) \vee \neg R(w,y))))$:
  $\equiv \exists x(\neg R(x,y) \vee \forall y(P(z,y) \wedge \exists w(S(w,u) \vee \neg R(w,y))))$
  $\equiv \exists x(\neg R(x,y) \vee \forall m(P(z,m) \wedge \exists w(S(w,u) \vee \neg R(w,m))))$
  $\equiv \exists x \forall m(\neg R(x,y) \vee (P(z,m) \wedge \exists w(S(w,u) \vee \neg R(w,m))))$
  $\equiv \exists x \forall m(\neg R(x,y) \vee \exists w(P(z,m) \wedge (S(w,u) \vee \neg R(w,m))))$
  $\equiv \underline{\exists x} \forall m \underline{\exists w}(\neg R(x,y) \vee (P(z,m) \wedge (S(w,u) \vee \neg R(w,m))))$ **FNP**
  $\exists x \Rightarrow$ adicionando uma constante de Skolem a na assinatura;
  $\exists w \Rightarrow$ adicionando uma função unária de Skolem $f(-)$ na assinatura;
  $\equiv \forall m(\neg R(a,y) \vee (P(z,m) \wedge (S(f(m),u) \vee \neg R(f(m),m))))$ **FPS**

Assim, para colocar as fórmulas na forma normal conjuntiva, primeiro colocamos-as na forma padrão de Skolem e simplesmente ignoramos os quantificadores universais.

Exercícios Resolvidos

1. Dadas as sentenças a seguir:

   1: Se o unicórnio é lenda, é imortal, mas se não é lenda, é mamífero.

   2: O unicórnio, se é imortal ou mamífero, é chifrudo.

   3: O unicórnio, se é chifrudo, é bruxaria.

   4: O unicórnio é bruxaria.

   5: O unicórnio é chifrudo.

   Prove que $\{1, 2, 3\} \vDash 4 \wedge 5$.

   Esse é o mesmo conjunto de sentenças visto no início da 2ª parte, e como vimos, estas são as sentenças na lógica de primeira ordem:

   1: $(L(u) \rightarrow I(u)) \wedge (\neg L(u) \rightarrow M(u))$

   2: $(I(u) \vee M(u)) \rightarrow C(u)$

   3: $C(u) \rightarrow B(u)$

   4: $B(u)$

   5: $C(u)$

   Queremos provar que $\{1, 2, 3\} \vDash 4 \wedge 5$. Sabemos que Resolução não responde se uma fórmula é consequência lógica de um conjunto de premissas, então utilizaremos a equivalência:

   $$\Gamma \vDash \varphi \text{ se, e somente se, } \Gamma \cup \{\neg\varphi\} \text{ é insatisfatível}$$

   Assim, queremos responder se $\{1, 2, 3, \neg(4 \wedge 5)\}$ é insatisfatível. Vamos transformar cada sentença do conjunto para a sua forma normal conjuntiva:

   - $(L(u) \rightarrow I(u)) \wedge (\neg L(u) \rightarrow M(u))$
     $\equiv (\neg L(u) \vee I(u)) \wedge (L(u) \vee M(u))$ **FNC**
   - $(I(u) \vee M(u)) \rightarrow C(u)$
     $\equiv \neg(I(u) \vee M(u)) \vee C(u)$
     $\equiv (\neg I(u) \wedge \neg M(u)) \vee C(u)$
     $\equiv (\neg I(u) \vee C(u)) \wedge (\neg M(u) \vee C(u))$ **FNC**
   - $C(u) \rightarrow B(u)$
     $\equiv \neg C(u) \vee B(u)$ **FNC**
   - $\neg(B(u) \wedge C(u))$
     $\equiv \neg B(u) \vee \neg C(u)$ **FNC**

   Assim:

   $$(\neg L(u) \vee I(u))^{C_1} \wedge (L(u) \vee M(u))^{C_2} \wedge (\neg I(u) \vee C(u))^{C_3} \wedge (\neg M(u) \vee C(u))^{C_4} \wedge (\neg C(u) \vee B(u))^{C_5} \wedge (\neg B(u) \vee \neg C(u))^{C_6}$$

   C_5 e $C_6 \Rightarrow \neg C(u)^{C_7}$

   C_7 e $C_4 \Rightarrow \neg M(u)^{C_8}$

   C_8 e $C_2 \Rightarrow L(u)^{C_9}$

   C_9 e $C_1 \Rightarrow I(u)^{C_{10}}$

   C_{10} e $C_3 \Rightarrow C(u)^{C_{11}}$

   C_{11} e $C_7 \Rightarrow ()$

   Encontrada a cláusula vazia, temos que $\{1, 2, 3, \neg(4 \wedge 5)\}$ é insatisfatível e, portanto, $\{1, 2, 3\} \vDash 4 \wedge 5$.

2. Prove que se uma relação binária é reflexiva e circular, então ela é simétrica.

Iremos tomar o símbolo R para representar a relação. Escrevendo as definições de relação reflexiva, circular e simétrica em lógica de primeira ordem, temos:

1: $\forall x R(x,x)$

2: $\forall x \forall y \forall z((R(x,y) \land R(y,z) \to R(z,x)))$

3: $\forall x \forall y(R(x,y) \to R(y,x))$

Queremos provar que $\{1,2\} \vDash 3$. Usaremos a equivalência:

$$\Gamma \vDash \varphi \text{ se, e somente se, } \Gamma \cup \{\neg\varphi\} \text{ é insatisfatível}$$

Ou seja, queremos provar que $\{1,2,\neg 3\}$ é insatisfatível. Transformando cada sentença para sua forma normal conjuntiva:

- $\forall x R(x,x)$
  $\equiv R(x,x)$ **FNC**

- $\forall x \forall y \forall z((R(x,y) \land R(y,z)) \to R(z,x))$
  $\equiv (R(x,y) \land R(y,z)) \to R(z,x)$
  $\equiv \neg(R(x,y) \land R(y,z)) \lor R(z,x)$
  $\equiv \neg R(x,y) \lor \neg R(y,z) \lor R(z,x)$ **FNC**

- $\neg\forall x \forall y(R(x,y) \to R(y,x))$
  $\equiv \exists x \exists y \neg(R(x,y) \to R(y,x))$ **FNP**
  $\exists x \Rightarrow$ adicionando uma constante de Skolem a na assinatura;
  $\exists y \Rightarrow$ adicionando uma constante de Skolem b na assinatura;
  $\equiv \neg(R(a,b) \to R(b,a))$ **FPS**
  $\equiv \neg(\neg R(a,b) \lor R(b,a))$
  $\equiv R(a,b) \land \neg R(b,a)$ **FNC**

Assim:

$$R(x,x)^{C_1} \land (\neg R(x,y) \lor \neg R(y,z) \lor R(z,x))^{C_2} \land R(a,b)^{C_3} \land \neg R(b,a)^{C_4}$$

C_1 e $C_4 \Rightarrow$ não são unificáveis

C_2 e C_3, pela unificadora $\Theta = [a / x, b / y] \Rightarrow (\neg R(b,z) \lor R(z,a))^{C_5}$

C_5 e C_4, pela unificadora $\Theta = [b / z] \Rightarrow \neg R(b,b)^{C_6}$

C_6 e C_1, pela unificadora $\Theta = [b / x] \Rightarrow ()$

Encontrada a cláusula vazia, temos que $\{1,2,\neg 3\}$ é insatisfatível e, portanto, $\{1,2\} \vDash 3$.

Exercícios

1. Para cada uma das fórmulas abaixo, transforme-a para a sua forma normal prenex e, em seguida, aplique a skolemização na fórmula resultante para obter a sua forma padrão de Skolem.

   (a) $\exists x(B(x) \land \forall y K(x,y)) \to \forall z(T(z) \land \neg\forall x B(x))$

   (b) $\forall x \exists y(\exists w(P(w) \land \neg Q(y)) \to \exists w \forall k(T(k,x) \lor P(w)))$

   (c) $\forall y(\forall x P(x,y) \to \exists z Q(x,z))$

   (d) $\exists x \forall y P(x,y) \lor \neg\exists y(Q(y) \to \forall z R(z))$

2. Construa uma estrutura que sirva como contraexemplo para mostrar que $\exists x P(x) \land \exists x Q(x) \not\equiv \exists x(P(x) \land Q(x))$.

3. Construa uma estrutura que sirva como contraexemplo para mostrar que $\forall x P(x) \lor \forall x Q(x) \not\equiv \forall x(P(x) \lor Q(x))$.

4. Use o algoritmo de Herbrand para determinar se os conjuntos de termos abaixo são unificáveis. Em caso positivo, dê a substituição unificadora mais geral.

   (a) $\{h(f(g(y))), h(f(g(g(a)))), h(y)\}$

   (b) $\{q(z,x,f(y)), q(z,h(z,w),f(w)), q(z,h(a,g(b)),f(g(b)))\}$

   (c) $\{h(f(x,y),g(f(a,y))), h(f(a,y),g(z)), h(f(a,b),g(w))\}$

   (d) $\{f(g(x,h(z))), f(g(x,h(x))), f(g(a,h(a)))\}$

5. Execute o Método da Resolução para responder às instâncias do Problema da Satisfatibilidade a seguir.

   (a) $\{P(x,a) \lor Q(x), \neg P(f(y),a) \lor Q(y), \neg Q(f(b))\} \vDash Q(b)$?

   (b) $\{\forall x(P(x) \to \exists y Q(x,y)), \forall x \forall y \forall z(Q(x,z) \to P(f(y)))\} \vDash \forall x(P(x) \to \exists y P(f(y)))$?

   (c) $\{\forall x(\exists y(S(x,y) \land M(y)) \to \exists y(I(y) \land E(x,y)))\} \vDash \neg\exists x I(x) \to \forall x \forall y(S(x,y) \to \neg M(y))$?

6. $\star$ Crie uma assinatura e uma estrutura apropriadas para representar as sentenças a seguir, e use o Método da Resolução para mostrar que $\{1,2,3,4,5\} \vDash 6$, onde:

   1: Todo cachorro uiva à noite.

   2: Qualquer um que tenha gato em casa não tem nenhum rato em casa.

   3: Todos que têm sono leve, não têm animal de estimação que uiva à noite em casa.

   4: João tem um animal de estimação em casa.

   5: Todo animal de estimação é um gato ou um cachorro.

   6: Se João tem sono leve, então João não tem ratos em casa.

8 A Incompletude da Matemática

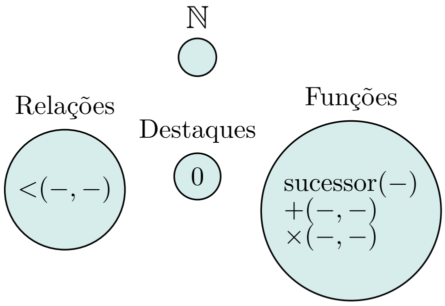
Vimos até agora as grandes potencialidades da lógica simbólica na resolução de validade de argumentos, consistência de um conjunto de sentenças, satisfatibilidade... Porém, a lógica simbólica têm limites para o que pode representar ou resolver. Observamos no capítulo anterior a conclusão de Alan Turing de que a Lógica de Primeira Ordem não é decidível (ou seja, não existe um procedimento confiável que nos permite dizer se qualquer fórmula é satisfatível ou não). Veremos nesse capítulo uma outra conclusão ainda mais intrigante, a qual chegou o matemático austríaco Kurt Gödel: na matemática, há enunciados verdadeiros para os quais não há prova.

8.1 O Programa de Hilbert

No fim do século XIX, os matemáticos estavam em busca de fundamentos para a matemática, ou seja, um conjunto contendo as definições mais elementares da matemática, que naquele momento, era tida como a Aritmética. Alguns matemáticos (incluindo Gottlob Frege, o fundador da Lógica de Primeira Ordem) tentaram definir tais fundamentos, mas foram pegos de surpresa com a aparição de paradoxos, que causou o questionamento dos métodos e lógica usada pela matemática. Esse período foi conhecido como a *crise dos fundamentos da matemática*.

Em resposta, o alemão David Hilbert propôs uma solução bastante ambiciosa: representar cada uma das teorias da matemática por um conjunto finito de axiomas (leis básicas), e provar que esse conjunto é consistente usando a lógica de primeira ordem. Ao fazermos isso, **poderíamos então provar qualquer enunciado verdadeiro sobre essa teoria usando os axiomas**.

Conjuntos desse tipo são ditos **teorias axiomáticas**. Por exemplo, em 1889, Giuseppe Peano propôs uma assinatura e uma teoria axiomática para a Aritmética consistindo de 7 leis básicas:



1 símbolo de destaque: 0;
1 símbolo de relação binária: $<$;
1 símbolo de função unária: s;
2 símbolos de função binária: $+$, $\times$

1. **0 não é sucessor de nenhum número.**
   $\neg\exists x(0 = s(x))$

2. **A função sucessor é injetora.**
   $\forall x\forall y((s(x) = s(y)) \to x = y)$

3. **Lei da Indução Matemática**
   $(P(0) \land \forall x(P(x) \to P(s(x)))) \to \forall y P(y)$

4. **Lei da Recursividade da Adição**
   $\forall x\forall y((x + s(y)) = s(x + y))$

5. **0 é elemento neutro da adição.**
   $\forall x((x+0)=x)$

6. **Lei da Recursividade da Multiplicação**
   $\forall x \forall y((x \times s(y)) = (x \times y) + x)$

7. **A multiplicação de qualquer número por 0 resulta em 0.**
   $\forall x((x \times 0) = 0)$

Podemos provar vários enunciados a partir desses axiomas, e então usá-los para provar mais enunciados. Por exemplo, podemos provar que 1 é elemento neutro da multiplicação usando os axiomas 5, 6 e 7:

$$\{\forall x((x+0)=x), \forall x \forall y((x \times s(y)) = (x \times y) + x), \forall x((x \times 0) = 0)\} \vdash \forall x((x \times s(0)) = x)$$

E essa é a proposta de Hilbert. Uma vez que isso possa ser feito, teremos um sistema que é **completo** (todo enunciado verdadeiro é demonstrável a partir dos axiomas) e **correto** (todo enunciado que é demonstrável a partir dos axiomas é verdadeiro).

8.2 O Teorema da Incompletude

Contudo, em 1931, Kurt Gödel deu uma “sentença de morte” ao programa. Ele usou a Aritmética de Peano para provar seu primeiro **Teorema da Incompletude**:

Teorema 8.2.1: Primeiro Teorema da Incompletude

Qualquer teoria axiomática que se proponha a formalizar a Aritmética não pode ser completa e correta ao mesmo tempo.

O teorema implica que, se supormos que tudo que demonstramos é verdadeiro (corretude), então há enunciados verdadeiros na matemática que não possuem prova (não completude). Por outro lado, se supormos que podemos provar qualquer enunciado verdadeiro (completude), então também podemos provar enunciados falsos (não corretude).

8.2.1 A Estratégia de Gödel

A ideia inusitada de Gödel foi usar a Aritmética para falar de si própria: ele propôs uma maneira de escrever quaisquer sentenças da Aritmética sob a forma de um natural. Para isso, ele atribuiu um número primo a todo símbolo da assinatura de Peano e da lógica de primeira ordem. Por exemplo:

0	$<$	s	$+$	$\times$	$\forall$	$\exists$	$\neg$	$\wedge$	$\vee$	$\rightarrow$	x	y	$=$	(	)	...
2	3	5	7	11	13	17	19	23	29	31	37	41	43	47	53	

Assim, ele propôs escrever quaisquer enunciados da Aritmética, verdadeiros ou falsos, sob a forma de produtos de números primos, usando a codificação acima como os seus expoentes. Por exemplo:

$$\neg \exists x(0 = s(x)) \equiv 2^{19} \times 3^{17} \times 5^{37} \times 7^{47} \times 11^{2} \times 13^{43} \times 17^{5} \times 19^{47} \times 23^{37} \times 29^{53} \times 31^{53}$$

$$\forall x((x+0)=0) \equiv 2^{13} \times 3^{47} \times 5^{47} \times 7^{37} \times 11^{7} \times 13^{2} \times 17^{53} \times 19^{43} \times 23^{2} \times 29^{53}$$

O Teorema Fundamental da Aritmética garante que cada número possui uma fatoração prima única, então a codificação de Gödel é válida.

O Paradoxo do Mentiroso

O Paradoxo do Mentiroso é a seguinte sentença: “Eu não sou verdadeira”. É um paradoxo pois, se supormos verdadeira, ela se afirma falsa e, se supormos falsa, ela se afirma verdadeira. Gödel tomou uma variante do paradoxo:

Eu não sou demonstrável a partir dos axiomas.

A sentença de Gödel não é um paradoxo, mas serviu para seu propósito. Gödel conseguiu codificar a sentença como um natural, usando a codificação anterior, implicando que ela é formalmente um enunciado da Aritmética. Tal feito é intrigante, pois:

- Supondo que a sentença é falsa, temos que ela é demonstrável a partir dos axiomas. Supondo que a Aritmética é correta, a sentença deve ser verdadeira, o que é uma contradição. Desse modo, a suposição de que ela é falsa não pode ser verdade, e portanto, a sentença deve ser verdadeira.
- Uma vez que é verdadeira, ela se afirma não demonstrável a partir dos axiomas. Assim, há uma sentença verdadeira na matemática que não podemos provar verdadeira, implicando que a Aritmética não é completa.
- Podemos fazer o mesmo raciocínio com a suposição de que a Aritmética é completa. Sendo assim, a sentença não pode ser verdadeira. Desse modo, há uma sentença demonstrável na Aritmética, mas que é falsa, implicando que a Aritmética não é correta.

Assim, a Aritmética não pode ser correta e completa ao mesmo tempo.