Apostila de Lógica

Lógica para Computação - IF673

João Victor de Sá Ferraz Coutinho

Este material é muito fortemente baseado nas aulas dos professores **Ruy José Guerra Barreto de Queiroz** e **Anjolina Grisi de Oliveira**, que ministram a disciplina Lógica para Computação (IF673). Eles possuem direitos sobre este material.

Julho, 2017

Sumário

Introdução à Lógica

1. Lógica Aristotélica

A Lógica surgiu a cerca de 350 a.C., com os estudos de Aristóteles. Ele buscou elencar as bases para a chamada Ciência da Argumentação: o estudo da forma dos argumentos.

1.1 Argumento

Suscintamente, um argumento é um conjunto de sentenças declarativas ou enunciados, que, por sua vez, são aquelas que admitem um **valor-verdade** — podem ser verdadeiras ou falsas, e são constituídas de termos que representam objetos (ou indivíduos) e termos que representam categorias (ou coleções).

1.1.1 Sentença declarativa

Segundo Aristóteles, existem dois tipos de sentenças declarativas:

1. As que relacionam objetos a categorias:

   x é P — x é um objeto e P uma categoria.
   x não é P

2. As que relacionam categorias a categorias:

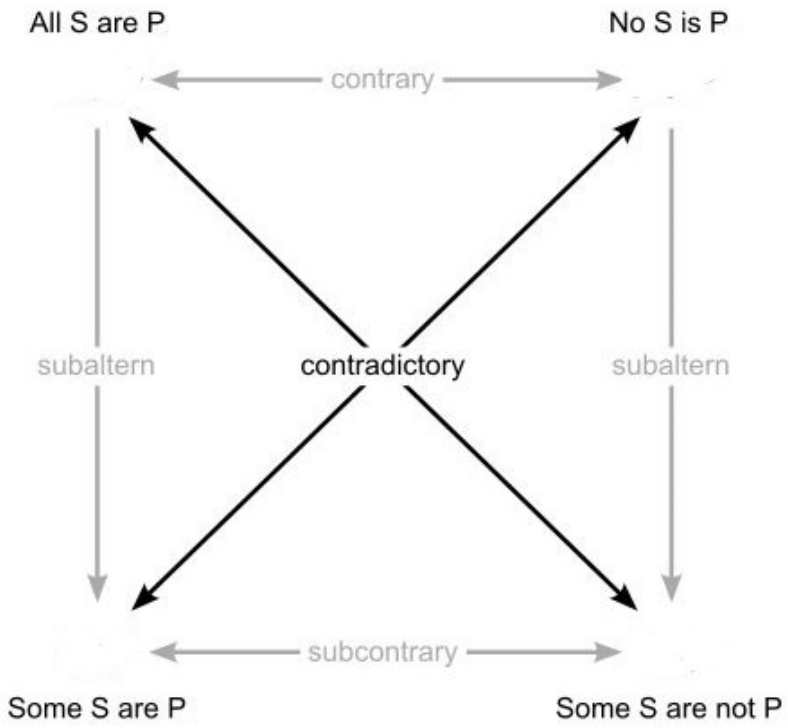
   Todo P é Q — universal positiva
   Nenhum P é Q — universal negativa
   Algum P é Q — existencial positiva
   Algum P não é Q — existencial negativa

Perceba duas coisas: primeiro, nós podemos representar objetos e categorias como elementos e conjuntos. Assim, "x é P"pode ser interpretado como $x \in P$, bem como "Todo P é Q"pode ser interpretado como $P \subseteq Q$. Segundo, note que é impossível x ser P e não ser P ao mesmo tempo. Ao afirmarmos as duas sentenças, encontramos um erro lógico. Esse erro chama-se contradição.

1.2 Contradição

Aristóteles identificou um requisito fundamental que deve ser atendido por todo bom argumento: a ausência de contradição. Quando combinações de sentenças declarativas levam a um absurdo ou impossibilidade lógica, temos uma contradição. Perceba que, para sentenças do tipo (1), ela é óbvia, afinal, x ser P e não ser P ao mesmo tempo é impossível. O mesmo não acontece para as sentenças do tipo (2). Para lidar com estas, Aristóteles usou o Quadrado das Oposições:

1.2.1 O Quadrado das Oposições



As Leis do Quadrado

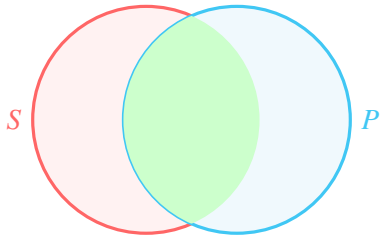
As arestas do quadrado são chamadas de leis. Sentenças:

Contrárias não podem ser verdadeiras juntas, mas podem ser falsas juntas.

Subcontrárias não podem ser falsas juntas, mas podem ser verdadeiras juntas.

Contraditórias não podem ter o mesmo valor-verdade (nem verdadeiras nem falsas juntas).

Subalternas estabelecem a relação "se o universal é verdadeiro, o existencial também é".



O diagrama acima ilustra as três primeiras leis.

1.3 Validez

1.3.1 Ato de Inferência

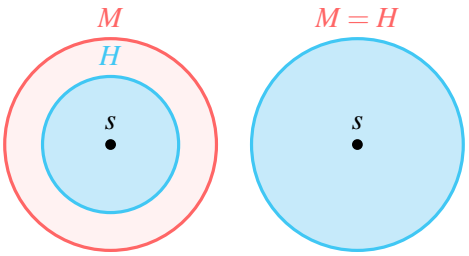
Além dos elementos básicos de um argumento definidos por Aristóteles, surge a necessidade de caracterizá-lo como algo não apenas descritivo. Isso é possível ao se verificar que um argumento envolve um ou mais **atos de inferência** a partir de sentenças tomadas como **premissas**, que permitem tirar **conclusões**.

Premissa 1
Premissa 2
...
Premissa n
---
logo, Conclusão

Dizemos que um ato de inferência é **válido** se se todas as suas premissas forem verdadeiras, a conclusão também for. Quando isso acontece, dá-se a ele o nome de silogismo.

■ **Exemplo 1.1** Determine a validez do ato de inferência a seguir:

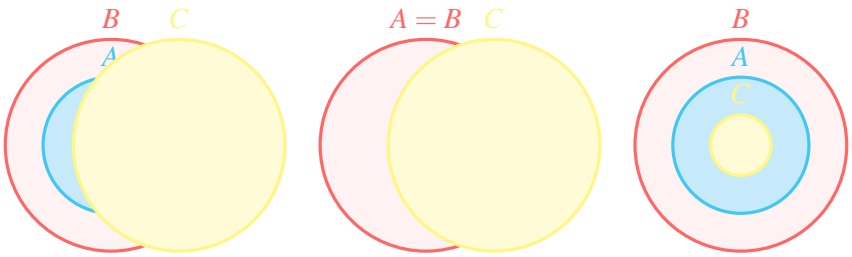
Todo H é M.
s é H.
---
logo, s é M.



Como, em ambas as configurações possíveis para as premissas, a conclusão é verdadeira, o ato é **válido**. ■

■ **Exemplo 1.2** Determine a validez do ato de inferência a seguir:

Todo A é B.
Algum B não é C.
---
logo, Algum C não é A.



Perceba que os três diagramas mostram uma configuração correta das premissas, mas o terceiro mostra uma conclusão contraditória à esperada. Assim, temos um ato de inferência **inválido**, pois há um caso em que todas as premissas são verdadeiras mas a conclusão não é. ■

Agora, temos o que precisamos para definir apropriadamente argumento e sua validez.

Definição 1.3.1 — Argumento. Um argumento é uma coleção de atos de inferência, e ele é válido se todos os seus atos também são — de modo que, se suas premissas forem verdadeiras, a conclusão final também é.

1.4 Consistência

Considere o seguinte conjunto de sentenças (vamos analisá-lo sem os diagramas dessa vez, mas fique livre para testá-los aqui):

Todo A é B.
Todo B é D.
Algum A não é D.

Se todo A é B, temos que $A \subseteq B$. Se todo B é D, temos que $B \subseteq D$. Por propriedade de conjuntos, concluímos seguramente que $A \subseteq D$. Porém, temos uma terceira sentença, "Algum A não é D", que nos diz que $A \nsubseteq D$. Como $A \subseteq D$ e $A \nsubseteq D$ é impossível, temos uma contradição. Quando isso ocorre, dizemos que o conjunto de sentenças é inconsistente.

Definição 1.4.1 — Consistência (definição formal). Um conjunto de sentenças é dito **inconsistente** se existe um ato de inferência baseado em algum subconjunto desse conjunto que deduz validamente uma sentença e, com base em outro subconjunto desse conjunto, deduz validamente uma sentença contraditória a anterior. O conjunto é consistente caso contrário.

Uma consequência direta da definição é que as sentenças do conjunto não terão o mesmo valor-verdade quando juntas. Como não estamos interessados em sentenças falsas — para premissas falsas, temos argumentos **válidos por vacuidade** —, podemos definir informalmente consistência da seguinte forma:

Definição 1.4.2 — Consistência (definição informal). Um conjunto de sentenças é dito **consistente** se existe uma configuração em que todas elas são verdadeiras ao mesmo tempo.

■ **Exemplo 1.3** Determine se o seguinte conjunto de sentenças é consistente.

Todo A é B.
Todo B é C.
Todo A é D.
Nenhum D é C.

Como todo A é B, $A \subseteq B$. Como todo B é C, $B \subseteq C$. Por propriedade de conjuntos, temos que $A \subseteq C$. Como todo A é D, $A \subseteq D$. Desse modo, como $A \subseteq C$ e $A \subseteq D$, concluímos seguramente que A é uma interseção entre C e D. Porém, como nenhum D é C, temos que $D \cap C = \varnothing$. Temos uma contradição, tornando o conjunto **inconsistente**.

Perceba que, ao usarmos diagramas de Venn, teríamos que mostrar que, para todas as configurações possíveis para as sentenças, nenhuma delas é verdade. ■

■ **Exemplo 1.4** Tome as sentenças do exemplo anterior como premissas e a sentença "Algum A não é B"como conclusão. Determine a validez do argumento.

Apesar de termos uma contradição direta entre a primeira premissa, "Todo A é B", e a conclusão, "Algum A não é B", o conjunto de premissas é inconsistente. Isso significa que não conseguimos avaliar se a conclusão é legítima ou não. Desse modo, o argumento é **válido por vacuidade**. ■

1.5 Restrição da Lógica Aristotélica

A lógica antiga não era plenamente formal, pois não era apática aos conteúdos das sentenças nem ao conhecimento dos sujeitos. Ou seja, era atribuida a uma sentença um valor-verdade baseado na falsidade ou verdade do conhecimento do sujeito. Desse modo, ambiguidades, equívocos e falta de clareza eram problemas enfrentados frequentemente pelos lógicos. Em oposição à essa linha de pensamento, surgiu independentemente ao estudo tradicional da lógica e como um sub-ramo da Matemática, a Lógica Simbólica. Sentenças e coleções de sentenças agora podiam ser representadas por símbolos, em um modelo matemático formal.

II Lógica Proposicional

2. Sintaxe

Estudaremos inicialmente o processo de formação e a forma de expressões da lógica — a **sintaxe**. Precisamos, portanto, definir um alfabeto (todos os símbolos que podemos usar para gerar uma expressão).

2.1 O Alfabeto da Lógica

Nosso alfabeto $\sum$ é composto por:

Variáveis: $x, y, z, w...$

Operadores: $\wedge, \vee, \neg, \rightarrow$

Constantes: $0, 1$

Parênteses: $(,)$

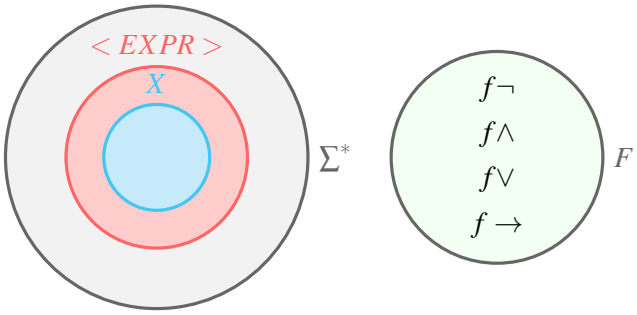
Palavras que podem ser formadas por esse alfabeto são, por exemplo:

$$(x \wedge y) \vee \neg w$$
$$(z \wedge 0) \rightarrow 1$$
$$0 \rightarrow \neg\neg))$$
$$(((()xw \vee \wedge$$

Podemos formar infinitas palavras com $\sum$, e então definimos $\sum^*$ como o conjunto de todas as palavras sobre o alfabeto.

As duas últimas palavras parecem um tanto estranhas, não? De fato, elas não **expressões legítimas**. Podemos definir assim o conjunto $EXPR$ como o conjunto de todas as expressões legítimas da lógica. Além disso, vamos precisar de um conjunto menor, que contenha as constantes e as variáveis — a nossa **base** X, e um conjunto de **funções** F, no qual seus elementos são representados pelos operadores.

Assim, temos algo semelhante a isto:



2.1.1 O Conjunto de Funções

Cada operador possui uma respectiva função presente em F com sua própria aridade.

$f\neg\colon \sum^* \mapsto \sum^*$
$f\neg(\omega) = \neg\omega$

$f\wedge\colon \sum^* \times \sum^* \mapsto \sum^*$
$f\wedge(\omega_1, \omega_2) = \omega_1 \wedge \omega_2$

$f\vee\colon \sum^* \times \sum^* \mapsto \sum^*$
$f\vee(\omega_1, \omega_2) = \omega_1 \vee \omega_2$

$f\rightarrow\colon \sum^* \times \sum^* \mapsto \sum^*$
$f\rightarrow(\omega_1, \omega_2) = \omega_1 \rightarrow \omega_2$

2.2 Conjuntos Indutivos

Como podemos definir o conjunto das expressões legítimas? Precisamos fazer isso **indutivamente**:

- toda variável é uma expressão legítima;
- toda constante é uma expressão legítima;
- se ω for uma expressão legítima, então $\neg\omega$ também é;
- se ω_1 e ω_2 forem expressões legítimas, então $(\omega_1 \wedge \omega_2)$ também é;
- se ω_1 e ω_2 forem expressões legítimas, então $(\omega_1 \vee \omega_2)$ também é;
- se ω_1 e ω_2 forem expressões legítimas, então $(\omega_1 \rightarrow \omega_2)$ também é;

Esse conjunto é um exemplo de **conjunto indutivo**, no qual a base, que possui certa característica, é "expandida"em um conjunto pela aplicação de funções nesse conjunto. Assim, generalizando nossa definição:

Definição 2.2.1 — Conjunto Indutivo. Seja A um conjunto qualquer e seja X um subconjunto próprio de A. Seja F um conjunto de funções sobre A, cada uma com sua aridade.
Um subconjunto Y de A é dito **indutivo sobre X e F** se:

1. Y contém X;
2. Y é fechado sobre as funções de F.

Note que o menor conjunto indutivo sobre X e F, na lógica, é o EXPR, enquanto o maior é o próprio $\sum^*$.

■ **Exemplo 2.1** Defina indutivamente o conjunto dos números naturais pares.

- Domínio $= \mathbb{N}$
- Base $X = \{0\}$
- Conjunto de funções $F = \{\ f(-)\ \}$, onde
  $f : \mathbb{N} \mapsto \mathbb{N}$
  $f(x) = x + 2$

Fazendo um paralelo com a definição, A é o domínio ($\mathbb{N}$), Y é o conjunto desejado (o conjunto dos naturais pares), X é a base e F o conjunto de funções sobre A. Infinitas aplicações sucessivas de $f(-)$ resultam no conjunto dos números naturais pares. ■

2.3 Fecho Indutivo

Assim como, em Teoria das Relações, possuímos fecho simétrico, transitivo e reflexivo de uma relação R em um conjunto B (a menor relação com a propriedade em B que contém R), também possuímos fecho indutivo sobre X e F: o menor conjunto que é indutivo sobre X e F. Para achá-lo, dispomos de dois métodos.

2.3.1 O Método Up-Down

Para encontrar o menor conjunto indutivo sobre X e F, tomemos a interseção de todos os subconjuntos de A que são conjuntos indutivos sobre X e F — como o próprio A é indutivo, a interseção não é vazia. Teremos, portanto, o **menor conjunto indutivo sobre X e F** (e sua notação é X^+).

$$X^+ = \bigcap \text{conjuntos indutivos sobre X e F}$$

2.3.2 O Método Bottom-Up

A partir da base, começamos a fazer aplicações sucessivas das funções de F em uma relação de recorrência. O **fecho indutivo sobre X e F** é tomado a partir da união de todos os conjuntos da sequência (e sua notação é X_+).

- $X_0 = X$
- $X_{n+1} = X_n \cup \{f(\omega_1, \omega_2, ..., \omega_k) / f \in F, aridade(f) = k, (\omega_1, \omega_2, ..., \omega_k) \in X_n\}$

$$X_+ = \bigcup_{i=0}^{\infty} X_i$$

Lema 2.3.1 — $X_+ = X^+$**.** O fecho indutivo é o menor conjunto indutivo sobre X e F.

Demonstração. Mostraremos um esboço da demonstração.
Queremos provar que $X_+ \subseteq X^+$ e $X^+ \subseteq X_+$. A prova será direta.

- $X^+ \subseteq X_+$:
  Pela definição, X_+ é um conjunto indutivo. Desse modo, fez parte da interseção que formou X^+. Assim, $X^+ \subseteq X_+$.
- $X_+ \subseteq X^+$:
  Queremos provar que todos os conjuntos que participaram da união que formou X_+ são subconjuntos de X^+. Faremos uma indução.
  Passo base. $X_0 \subseteq X^+$. Como $X_0 = X$ e X^+ é indutivo, X^+ contém X.

Passo indutivo.
- Hipótese indutiva: $X_k \subseteq X^+$.
- Tese: $X_{k+1} \subseteq X^+$.
Aplicando as funções de F em X_k e unindo ao mesmo, obtemos $X_k \cup \{f(\omega_1, \omega_2, ..., \omega_n)/f \in F, aridade(f) = n, (\omega_1, \omega_2, ..., \omega_n) \in X_k\}$. Pela hipótese, $X_k \subseteq X^+$. Além disso, como X^+ é indutivo, é fechado sobre as funções de F, logo, a união ainda pertence ao conjunto. Desse modo, pela definição, temos que $X_{k+1} \subseteq X^+$, concluindo a tese.

Assim, $X_+ = X^+$. ■

2.4 Conjunto Livremente Gerado

Como dissemos anteriormente, desejamos caracterizar o conjunto de palavras sobre o alfabeto que são expressões da lógica proposicional (vamos nos referir a elas como **proposições** e ao conjunto *EXPR*, como *PROP*). Usando os conceitos de conjunto indutivo, somos capazes de:

- Reconhecer palavras sobre $\sum$ que são proposições.
- Aplicar funções recursivas sobre o conjunto das proposições para encontrar propriedades de sintaxe sobre seus elementos.
- Usar indução matemática para provar uma afirmação do tipo "todo elemento do conjunto das proposições tem propriedade X".

Para garantir que essas afirmações são seguras matematicamente, é necessário que cada proposição tenha apenas uma única linha de formação, ou seja, que o conjunto das proposições tenha a propriedade de **leitura única**. Assim, o conjunto precisa atender certas condições, sistematizadas no conceito a seguir.

Definição 2.4.1 — Conjunto Livremente Gerado. Seja A um conjunto qualquer e X um subconjunto próprio de A. Seja F um conjunto de funções sobre A, cada uma com sua aridade. O fecho indutivo X_+ de X sob F é dito **livremente gerado** se:

1. Todas as funções de F são injetoras;
2. Para quaisquer duas funções f e g de F, seus conjuntos imagem em relação a X_+ são disjuntos.
3. O conjunto imagem de qualquer função f de F em relação a X_+ não contém nenhum elemento da base X.

■ **Exemplo 2.2** O conjunto definido no exemplo 2.1 é livremente gerado?

1. f, a única função de F, é injetora.
2. F só possui uma função, então é garantido que não há outra com conjunto imagem não disjunto ao conjunto imagem de f.
3. O conjunto imagem de f, a única função de F, é $\{2,4,6,8,...\}$. Assim, não possui 0, que é o único elemento da base.

Atendendo às três condições, temos que o conjunto é livremente gerado. ■

2.4.1 Definindo funções recursivas sobre PROP

Podemos, agora que sabemos identificar expressões legítimas, definir funções recursivas que nos dão propriedades de uma proposição. Vamos passar por algumas delas.

Número de parênteses

$f : PROP \mapsto \mathbb{N}$
$f(\varphi) = 0$, se φ é atômica.
$f(\neg\psi) = f(\psi) + 2$.
$f(\rho\square\theta) = f(\rho) + f(\theta) + 2$, onde $\square \in \{\vee, \wedge, \rightarrow\}$.

Número de operadores

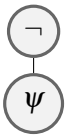
$g : PROP \mapsto \mathbb{N}$
$g(\varphi) = 0$, se φ é atômica.
$g(\neg\psi) = g(\psi) + 1$.
$g(\rho\square\theta) = g(\rho) + g(\theta) + 1$, onde $\square \in \{\vee, \wedge, \rightarrow\}$.

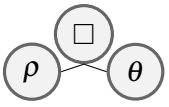
Árvore Sintática

$h : PROP \mapsto GRAFO$

$h(\varphi) =$ 

, se φ é atômica.

$h(\neg\psi) =$ 

$h(\rho\square\theta) =$ 

, onde $\square \in \{\vee, \wedge, \rightarrow\}$

Posto da Árvore Sintática

Poderíamos definir uma função $u : GRAFO \mapsto \mathbb{N}$, mas podemos ser mais práticos! Seja $p = u \circ h$.
$p : PROP \mapsto \mathbb{N}$
$p(\varphi) = 0$, se φ é atômica.
$p(\neg\psi) = p(\psi) + 1$.
$p(\rho\square\theta) = max(p(\rho), p(\theta)) + 1$, onde $\square \in \{\vee, \wedge, \rightarrow\}$.

Conjunto das Subexpressões

$s : PROP \mapsto \{PROP\}$
$s(\varphi) = \{\varphi\}$, se φ é atômica.
$s(\neg\psi) = s(\psi) \cup \{\neg\psi\}$.
$s(\rho\square\theta) = s(\rho) \cup s(\theta) \cup \{\rho\square\theta\}$, onde $\square \in \{\vee, \wedge, \rightarrow\}$.
Se quisermos o número de subexpressões, basta tomar a cardinalidade do conjunto.

■ **Exemplo 2.3** Determine o conjunto de subexpressões da proposição $((\neg((w \vee x)) \wedge z) \rightarrow 1)$.

Tomemos $s(((\neg((w \vee x)) \wedge z) \rightarrow 1))$.
$= s((\neg((w \vee x)) \wedge z)) \cup s(1) \cup \{((\neg((w \vee x)) \wedge z) \rightarrow 1)\}$
$= s(\neg((w \vee x))) \cup s(z) \cup \{(\neg((w \vee x)) \wedge z)\} \cup \{1\} \cup \{((\neg((w \vee x)) \wedge z) \rightarrow 1)\}$
$= s((w \vee x)) \cup \{\neg((w \vee x))\} \cup \{z\} \cup \{(\neg((w \vee x)) \wedge z), 1, ((\neg((w \vee x)) \wedge z) \rightarrow 1)\}$
$= s(w) \cup s(x) \cup \{((w \vee x))\} \cup \{\neg((w \vee x)), z, (\neg((w \vee x)) \wedge z), 1, ((\neg((w \vee x)) \wedge z) \rightarrow 1)\}$
$= \{w\} \cup \{x\} \cup \{((w \vee x)), \neg((w \vee x)), z, (\neg((w \vee x)) \wedge z), 1, ((\neg((w \vee x)) \wedge z) \rightarrow 1)\}$
$= \{w, x, ((w \vee x)), \neg((w \vee x)), z, (\neg((w \vee x)) \wedge z), 1, ((\neg((w \vee x)) \wedge z) \rightarrow 1)\}$. ■

2.4.2 Prova por indução sobre propriedades dos elementos de PROP

Finalmente, uma vez que sabemos definir funções que nos retornam propriedades sobre proposições, podemos usar indução para provar lemas do tipo "toda proposição tem a propriedade X".

Como proposições podem assumir 3 formas, precisamos fazer uma prova por indução **sobre a complexidade** da proposição. Vamos percorrer alguns exemplos.

Lema 2.4.1 Para toda proposição ϕ, o número de parênteses de ϕ é par.

Demonstração. Queremos provar que $f(\phi) = 2k$, onde $k \in \mathbb{Z}$.
Passo base. ϕ é atômica. Assim, $f(\phi) = 0 = 2 \times 0$, que é par.

Passo indutivo para ϕ da forma $\neg\psi$.
- Hipótese indutiva: $f(\psi) = 2k$.
- Tese: $f(\neg\psi) = 2k^{'}$.
Adicionando 2 em ambos os lados da hipótese, temos $f(\psi)+2 = 2k+2$.
Pela definição, $f(\neg\psi) = f(\psi)+2$ e, tomemos $k^{'} = k+1$.
Assim, temos $f(\neg\psi) = 2k^{'}$, concluindo a tese.

Passo indutivo para ϕ da forma $\rho\Box\theta$, onde $\Box \in \{\vee, \wedge, \rightarrow\}$.
- Hipótese indutiva: **(1)** $f(\rho) = 2k_1$; **(2)** $f(\theta) = 2k_2$.
- Tese: $f(\rho\Box\theta) = 2k^{'}$.
Somando as duas hipóteses, temos $f(\rho)+f(\theta) = 2k_1+2k_2$.
Adicionando 2 em ambos os lados da hipótese, temos $f(\rho)+f(\theta)+2 = 2k_1+2k_2+2$.
Pela definição, $f(\rho\Box\theta) = f(\rho)+f(\theta)+2$ e, tomemos $k^{'} = k_1+k_2+1$.
Assim, temos $f(\rho\Box\theta) = 2k^{'}$, concluindo a tese. ■

Lema 2.4.2 Para toda proposição ϕ, o número de subexpressões de ϕ é no máximo igual ao sucessor do dobro do número de operadores de ϕ.

Demonstração. Queremos provar que $|s(\phi)| \leq 2g(\phi)+1$.
Passo base. ϕ é atômica. Assim, $|s(\phi)| = |\{\phi\}| = 1$ e $g(\phi) = 0$. Temos, portanto, $1 \leq 1$.

Passo indutivo para ϕ da forma $\neg\psi$.
- Hipótese indutiva: $|s(\psi)| \leq 2g(\psi)+1$.
- Tese: $|s(\neg\psi)| \leq 2g(\neg\psi)+1$.
Adicionando 1 ao primeiro lado da hipótese e 2 ao segundo, temos $|s(\psi)|+1 \leq 2g(\psi)+1+2$.
Pela definição, $|s(\neg\psi)|$[1] $= |s(\psi)|+|\{\neg\psi\}| = |s(\psi)|+1$ *($s(\psi)$ e $\{\neg\psi\}$ são disjuntos).*
Como $g(\neg\psi) = g(\psi)+1$, temos que $2g(\neg\psi) = 2g(\psi)+2$.
Assim, temos $|s(\neg\psi)| \leq 2g(\neg\psi)+1$, concluindo a tese.

Passo indutivo para ϕ da forma $\rho\Box\theta$, onde $\Box \in \{\vee, \wedge, \rightarrow\}$.
- Hipótese indutiva: **(1)** $|s(\rho)| \leq 2g(\rho)+1$; **(2)** $|s(\theta)| \leq 2g(\theta)+1$.
- Tese: $|s(\rho\Box\theta)| \leq 2g(\rho\Box\theta)+1$.
Somando as duas hipóteses, temos $|s(\rho)|+|s(\theta)| \leq 2g(\rho)+2g(\theta)+2$.
Adicionando 1 a ambos os lados, temos $|s(\rho)|+|s(\theta)|+1 \leq 2g(\rho)+2g(\theta)+2+1$.
Pela definição, $|s(\rho\Box\theta)|$[2] $= |s(\rho)|+|s(\theta)|+|\{\rho\Box\theta\}| = |s(\rho)|+|s(\theta)|+1$.
Como $g(\rho\Box\theta) = g(\rho)+g(\theta)+1$, temos que $2g(\rho\Box\theta) = 2g(\rho)+2g(\theta)+2$.
Assim, temos $|s(\rho\Box\theta)| \leq 2g(\rho\Box\theta)+1$, concluindo a tese. ■

[1] $|A \cup B| = |A|+|B|-|A \cap B|$
[2] $|A \cup B \cup C| = |A|+|B|+|C|-(|A \cap B|+|A \cap C|+|B \cap C|+|A \cap B \cap C|)$

Lema 2.4.3 Para toda proposição ϕ, o posto da árvore sintática de ϕ é no máximo igual ao número de operadores de ϕ.

Demonstração. Queremos provar que $p(\phi) \leq g(\phi)$.
Passo base. ϕ é atômica. Assim, $p(\phi) = 0$ e $g(\phi) = 0$. Temos, portanto, $0 \leq 0$.

Passo indutivo para ϕ da forma $\neg\psi$.
- Hipótese indutiva: $p(\psi) \leq g(\psi)$.
- Tese: $p(\neg\psi) \leq g(\neg\psi)$.
Adicionando 1 em ambos os lados da hipótese, temos $p(\psi)+1 \leq g(\psi)+1$.
Pela definição, $p(\neg\psi) = p(\psi)+1$ e $g(\neg\psi) = g(\psi)+1$.
Assim, temos $p(\neg\psi) \leq g(\neg\psi)$, concluindo a tese.

Passo indutivo para ϕ da forma $\rho\Box\theta$, onde $\Box \in \{\vee, \wedge, \rightarrow\}$.
- Hipótese indutiva: **(1)** $p(\rho) \leq g(\rho)$; **(2)** $p(\theta) \leq g(\theta)$.
- Tese: $p(\rho\Box\theta) \leq g(\rho\Box\theta)$.
Somando as duas hipóteses, temos $p(\rho)+p(\theta) \leq g(\rho)+g(\theta)$.
Sabemos que $max(p(\rho), p(\theta)) \leq p(\rho)+p(\theta)$. Por transitividade, $max(p(\rho), p(\theta)) \leq g(\rho)+g(\theta)$.
Adicionando 1 aos dois lados da inequação, temos $max(p(\rho), p(\theta))+1 \leq g(\rho)+g(\theta)+1$.
Pela definição, $p(\rho\Box\theta) = max(p(\rho), p(\theta))+1$ e $g(\rho\Box\theta) = g(\rho)+g(\theta)+1$.
Assim, temos $p(\rho\Box\theta) \leq g(\rho\Box\theta)$, concluindo a tese. ■

3. Semântica

Agora que estudamos a forma de expressões da lógica, precisamos atribuir **significados** a elas — o estudo da **semântica**.

3.1 O Teorema da Extensão Homomórfica Única

Suponha as seguintes sentenças (como estamos na Lógica Simbólica, vamos colocá-las na forma de símbolos):

$x \equiv$ "O ano 2017 tem 365 dias."

$y \equiv$ "O ano 2013 é bissexto."

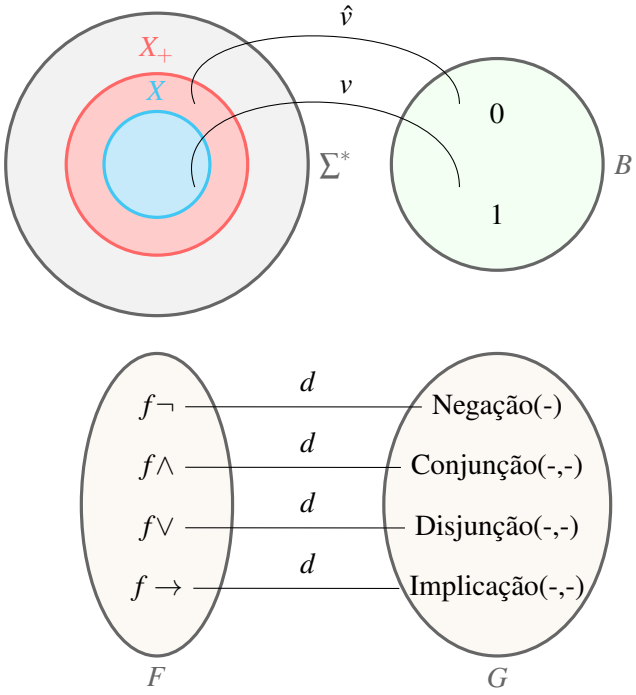
$z \equiv$ "O ano 2100 é múltiplo de 4."

Queremos saber o valor-verdade da proposição $((x \vee y) \rightarrow (y \wedge z))$. Perceba que ela de fato é uma expressão legítima da lógica, mas não conseguimos determinar seu valor-verdade imediatamente. O mesmo não ocorre com as variáveis: sabemos que os valores-verdade de x, y e z são **verdadeiro**, **falso** e **verdadeiro**, respectivamente. Assim, precisamos de um meio que nos permita **valorar** (atribuir valores-verdade a) expressões fora da base. É aí que entra o teorema a seguir.

Teorema 3.1.1 — Teorema da Extensão Homomórfica Única. Seja A um conjunto qualquer, X um subconjunto próprio de A e F um conjunto de funções sobre A, cada uma com sua aridade. Seja B um outro conjunto, G um conjunto de funções sobre B e seja $d : F \mapsto G$ um mapeamento entre as funções de F e G. Se o fecho indutivo de X sobre F for livremente gerado, então, para toda função $h : X \mapsto B$, existe uma única extensão $\hat{h} : X_+ \mapsto B$, de modo que:

1. $\hat{h}(\omega) = h(\omega)$, se $\omega \in X$;
2. Para toda função $f \in F / aridade(f) = k$, e para toda k-upla $(\omega_1, ..., \omega_k)$ de elementos de X_+, $\hat{h}(f(\omega_1, ..., \omega_k)) = g(\hat{h}(\omega_1), ..., \hat{h}(\omega_k))$, onde $g = d(f)$.

O teorema é generalizado para quaisquer conjuntos que atendam às suas condições. Contudo, por ora, vamos ver a consequência dele na Lógica Simbólica:



3.1.1 As Funções Booleanas

As funções de G na lógica são chamadas de funções booleanas. Elas agem sobre o conjunto B — o conjunto dos valores booleanos —, com 0 representando o valor-verdade falso, e 1, o verdadeiro.

Negação

A negação de um valor-verdade é o seu inverso.

$$\textbf{Negação}(\mathbf{0}) = \mathbf{1} \;\big|\; \textbf{Negação}(\mathbf{1}) = \mathbf{0}$$

Conjunção

A conjunção entre dois valores-verdade é verdadeira quando ambos são verdadeiros.

$$\textbf{Conjunção}(\mathbf{0},\mathbf{0}) = \mathbf{0} \;\big|\; \textbf{Conjunção}(\mathbf{0},\mathbf{1}) = \mathbf{0} \;\big|\; \textbf{Conjunção}(\mathbf{1},\mathbf{0}) = \mathbf{0} \;\big|\; \textbf{Conjunção}(\mathbf{1},\mathbf{1}) = \mathbf{1}$$

Disjunção

A disjunção entre dois valores-verdade é verdadeira quando pelo menos um deles é verdadeiro.

$$\textbf{Disjunção}(\mathbf{0},\mathbf{0}) = \mathbf{0} \;\big|\; \textbf{Disjunção}(\mathbf{0},\mathbf{1}) = \mathbf{1} \;\big|\; \textbf{Disjunção}(\mathbf{1},\mathbf{0}) = \mathbf{1} \;\big|\; \textbf{Disjunção}(\mathbf{1},\mathbf{1}) = \mathbf{1}$$

Implicação

A implicação entre dois valores-verdade é verdadeira quando o primeiro é falso ou o segundo é verdadeiro.

$$\textbf{Implicação}(\mathbf{0},\mathbf{0}) = \mathbf{1} \;\big|\; \textbf{Implicação}(\mathbf{0},\mathbf{1}) = \mathbf{1} \;\big|\; \textbf{Implicação}(\mathbf{1},\mathbf{0}) = \mathbf{0} \;\big|\; \textbf{Implicação}(\mathbf{1},\mathbf{1}) = \mathbf{1}$$

3.2 Valoração-Verdade

A função $\hat{v} : X_+ \mapsto B$ é a função de **valoração-verdade**, que é extensão de $v : X \mapsto B$. Ela é um caso especial do Teorema da Extensão Homomórfica Única, no qual **este nos permite calcular a valoração-verdade de uma proposição recursivamente, baseado no fato de que os valores-verdade das variáveis que nela ocorrem estão fixados**.

Definição 3.2.1 — Valoração-Verdade. A função de valoração-verdade é uma função $v : X \mapsto BOOLEANOS$, tal que sua extensão homomórfica $\hat{v} : X_+ \mapsto BOOLEANOS$ atende as condições:

- $\hat{v}(\omega) = v(\omega)$, se $\omega \in X$;
- $\hat{v}(f\neg(\omega)) = \text{Negação}(\hat{v}(\omega))$;
- $\hat{v}(f\wedge(\omega_1,\omega_2)) = \text{Conjunção}(\hat{v}(\omega_1),\hat{v}(\omega_2))$;
- $\hat{v}(f\vee(\omega_1,\omega_2)) = \text{Disjunção}(\hat{v}(\omega_1),\hat{v}(\omega_2))$;
- $\hat{v}(f\to(\omega_1,\omega_2)) = \text{Implicação}(\hat{v}(\omega_1),\hat{v}(\omega_2))$;

Seja ϕ uma proposição. Dizemos que uma dada valoração-verdade **satisfaz** ϕ se $\hat{v}(\phi) = 1$. Caso contrário, se $\hat{v}(\phi) = 0$, dizemos que a valoração-verdade **refuta** ϕ.

■ **Exemplo 3.1** Use valoração-verdade para valorar a proposição do exemplo inicial — $\phi = ((x\vee y)\to(y\wedge z))$, onde $v(x) = 1, v(y) = 0, v(z) = 1$.

Tomemos $\hat{v}(\phi) = \hat{v}(((x\vee y)\to(y\wedge z)))$.
$\hat{v}(\phi) = \text{Implicação}(\hat{v}((x\vee y)),\hat{v}((y\wedge z)))$
$\hat{v}(\phi) = \text{Implicação}(\text{Disjunção}(\hat{v}(x),\hat{v}(y)),\text{Conjunção}(\hat{v}(y),\hat{v}(z)))$
$\hat{v}(\phi) = \text{Implicação}(\text{Disjunção}(1,0),\text{Conjunção}(0,1))$
$\hat{v}(\phi) = \text{Implicação}(1,0) = 0$.

Nossa proposição é **falsa**, afinal, e a valoração-verdade $\{1, 0, 1\}$ a **refuta**. ■

3.3 Conjuntos de conectivos funcionalmente completos

Muitas vezes, desejamos escrever proposições que estão dadas sobre um conjunto de conectivos na forma de um outro conjunto de conectivos — geralmente menor, a fim de acelerar a computação —, mas que, no fim, as proposições sejam equivalentes. Por exemplo, o conjunto $\{\neg,\vee\}$ é funcionalmente completo, pois **temos uma equivalência a qualquer fórmula da lógica proposicional com apenas seus elementos**:

- $\neg\phi \equiv \neg\phi$
- $\phi\vee\psi \equiv \phi\vee\psi$
- $\phi\wedge\psi \equiv \neg(\neg\phi\vee\neg\psi)$ - *Lei de DeMorgan*
- $\phi\to\psi \equiv \neg\phi\vee\psi$

4. O Problema da Satisfatibilidade

Dada uma proposição ϕ, pergunta-se: ***ϕ é satisfatível?***

Responder à pergunta não é fácil. O problema da satisfatibilidade (também conhecido como SAT) é o precursor dos problemas NP-Completo: dada a solução pronta de uma instância do problema, podemos **verificá-la** (comprová-la) rapidamente, mas ainda não existem algoritmos rápidos o suficiente para efetivamente **encontrar** a solução — todos eles aumentam seu tempo muito depressa conforme a entrada aumenta. Contudo, heurísticas que resolvem o problema, atualmente, são capazes de resolver essas instâncias envolvendo fórmulas consistindo de milhões de símbolos. Vamos estudar algumas delas neste capítulo.

4.1 Satisfatibilidade

Tomemos alguns conceitos importantes.

φ é satisfatível se existe pelo menos uma valoração-verdade que satisfaz φ.

φ é refutável se existe pelo menos uma valoração-verdade que refuta φ.

φ é tautologia se todas as valorações-verdade satisfazem φ.

φ é insatisfatível se todas as valorações-verdade refutam φ.

φ e ψ são logicamente equivalentes se, para toda valoração-verdade v, $\hat{v}(\varphi) = \hat{v}(\psi)$.

Um conjunto Γ de proposições é satisfatível se existe pelo menos uma valoração-verdade que satisfaz todas as proposições de Γ.

φ é consequência lógica de Γ se todas as valorações-verdade que satisfazem todas as proposições de Γ também satisfazem φ (notação: $\Gamma \models \varphi$).

4.2 Método da Tabela-Verdade

Definido em 1921 pelo filósofo austríaco Ludwig Witigenstein, o método da tabela-verdade foi o primeiro método algorítmico que resolve SAT. Construímos uma tabela, onde suas linhas representam valores-verdade, e as colunas, as subexpressões (em ordem de complexidade) de uma proposição a qual desejamos analisar sua satisfatibilidade. Se, na última coluna, o valor 1 aparecer, a proposição é satisfatível.

■ **Exemplo 4.1** $(\neg(x \to (y \to \neg z)) \to y)$ é satisfatível?

x	y	z	$\neg z$	$(y \to \neg z)$	$x \to (y \to \neg z)$	$\neg(x \to (y \to \neg z))$	$(\neg(x \to (y \to \neg z)) \to y)$
0	0	0	1	1	1	0	1
0	0	1	0	1	1	0	1
0	1	0	1	1	1	0	1
0	1	1	0	0	1	0	1
1	0	0	1	1	1	0	1
1	0	1	0	1	1	0	1
1	1	0	1	1	1	0	1
1	1	1	0	0	0	1	1

Note que, além de ser safistatível, também é **tautologia**. ■

■ **Exemplo 4.2** Mostre que $\{\neg Q, (P \to Q)\} \models \neg P$ - *Contrapositividade*.

P	Q	$\neg P$	$\neg Q$	$(P \to Q)$
0	0	1	1	1
0	1	1	0	1
1	0	0	1	0
1	1	0	0	1

Como todas as valorações que satisfazem as duas proposições do conjunto (somente na primeira isso acontece) também satisfazem $\neg P$, temos que é consequência lógica. ■

4.2.1 Complexidade computacional

O método da tabela tem um problema muito grave: ele é o método que resolve SAT mais exaustivo que existe, pois verifica todos os valores-verdade possíveis para uma proposição. Vamos analisar isso computacionalmente. Dada uma entrada com n variáveis e k conectivos, a tabela tem 2^n linhas (o número de possibilidades para os valores-verdade das variáveis) e, no máximo, $2k+1$ colunas (provamos isso por indução lá atrás). Assim, são necessárias $2^n \times (2k+1)$ operações para montá-la. Isso quer dizer que **a complexidade computacional do método da tabela-verdade é da ordem de** $\Theta(2^n)$! Custos exponenciais são totalmente indesejáveis em computação, tornando o método inviável. Porém, se quisermos conferir se uma valoração-verdade satisfaz uma proposição, precisamos apenas olhar a linha dessa valoração, tornando o custo a apenas $2k+1$.

4.2.2 Corretude e completude

Apesar de seu custo grande, o método da tabela-verdade foi a base para todos os métodos seguintes, os quais tem que atender duas condições:

Corretude A resposta que o método fornece deve ser a mesma que o método da tabela-verdade forneceria.

Completude O método não deve prosseguir indefinidamente, ou seja, ele deve dar uma resposta com um tempo menor ou igual a $2^n \times (2k+1)$.

4.3 Método dos Tableaux Analíticos

Na década de 50, os lógicos Evert Beth (holandês) e Jaako Hintikka (finlandês) buscaram independentemente uma formulação intuitiva do conceito de valoração-verdade e encontraram uma noção filosófica de **mundo possível**. Assim, buscaram estabelecer uma ponte entre valoração-verdade e mundo possível e, então, formularam um método algorítmico que resolve SAT, que se baseia numa **árvore de possibilidades** — o método dos tableaux. Com ausência de contradição, **o mundo possível seria revelado por um caminho da raiz a uma folha**. Mais tarde, em 1970, a fim de melhorar o desempenho do método, o lógico americano Raymond Smullyan definiu uma metodologia para a aplicação de suas regras. A partir de então, o método ficou conhecido como o método dos tableaux analíticos.

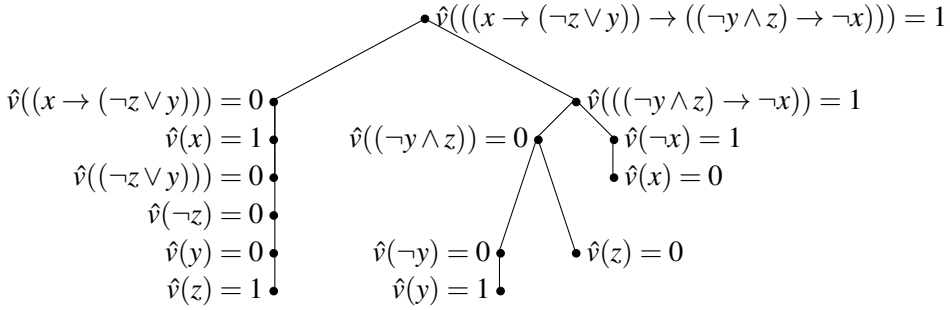
4.3.1 As regras do tableau

Temos um conjunto de 8 regras, divididas de acordo com os operadores e seu **tipo de bifurcação**: regras do tipo α não forçam a árvore a bifurcar, enquanto regras do tipo β bifurcam a árvore.

	α	β
$\neg$	$\hat{v}(\neg\psi)=1$ $\hat{v}(\psi)=0$ $\hat{v}(\neg\psi)=0$ $\hat{v}(\psi)=1$	
$\wedge$	$\hat{v}(\rho\wedge\theta)=1$ $\hat{v}(\rho)=1$ $\hat{v}(\theta)=1$	$\hat{v}(\rho\wedge\theta)=0$ $\hat{v}(\rho)=0$ $\quad$ $\hat{v}(\theta)=0$
$\vee$	$\hat{v}(\rho\vee\theta)=0$ $\hat{v}(\rho)=0$ $\hat{v}(\theta)=0$	$\hat{v}(\rho\vee\theta)=1$ $\hat{v}(\rho)=1$ $\quad$ $\hat{v}(\theta)=1$
$\rightarrow$	$\hat{v}(\rho\rightarrow\theta)=0$ $\hat{v}(\rho)=1$ $\hat{v}(\theta)=0$	$\hat{v}(\rho\rightarrow\theta)=1$ $\hat{v}(\rho)=0$ $\quad$ $\hat{v}(\theta)=1$

O tableau só pode responder se uma fórmula é satisfatível ou refutável. Dado uma instância do problema, construímos inicialmente uma árvore com as **condições iniciais** do tableau, aplicamos suas regras (dando preferência para as do tipo α) até encontrarmos as proposições atômicas e verificamos um caminho sem contradição da raiz às folhas. Caso haja, a resposta do método é "sim"e dizemos que o tableau está **aberto**. Caso contrário, a resposta do método é "não"e dizemos que o tableau está **fechado**.

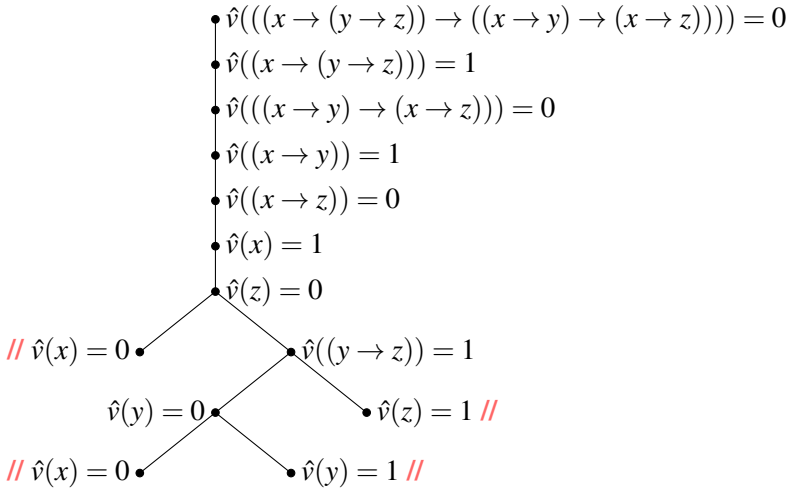
■ **Exemplo 4.3** $((x\rightarrow(\neg z\vee y))\rightarrow((\neg y\wedge z)\rightarrow\neg x))$ é satisfatível?



Como existe pelo menos um mundo possível, a resposta do método é **sim**. ■

■ **Exemplo 4.4** $((x \to (y \to z)) \to ((x \to y) \to (x \to z)))$ é tautologia?

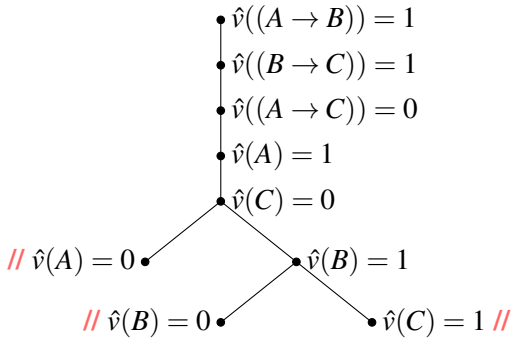
Aqui temos um problema, pois o método não responde se uma proposição é tautologia. Contudo, temos que **uma proposição φ é tautologia se, e somente se, φ não é refutável**. Assim, desejamos fazer com que o método responda **não** ao perguntarmos se a proposição é refutável.



Encontramos uma contradição em todos os ramos. Assim, o tableau está fechado e a resposta do método é **não**. Logo, a proposição **é tautologia**.
Perceba que existe mais de um modo de montar o tableau. Aqui, demos preferência para as regras que não bifurcam, diminuindo o número de operações (mas você poderia montá-lo em ordem de aparição dos vértices, por exemplo). Além disso, note que, ao encontrarmos uma contradição em um caminho, não precisamos mais seguir por ele — tal mundo não existe. ■

■ **Exemplo 4.5** Prove que $\{(A \to B), (B \to C)\} \models (A \to C)$ - *Transitividade da implicação*.

Precisamos que **não** exista um mundo onde as proposições do conjunto sejam verdadeiras e a de fora seja falsa (em outras palavras, o conjunto precisa ser satisfatível e a proposição, tautologia).



■

4.3.2 Complexidade computacional

Para alguns casos, o custo computacional do método dos tableaux analíticos se equipara ao da tabela-verdade — $O(2^n)$, tornando-o inviável nesses casos. Isso porque ocorrem muitas bifurcações e criações de novos vértices (que podem possivelmente também bifurcar). *Por exemplo, tente resolver "$(((x \wedge y) \vee (x \wedge \neg y)) \vee ((\neg x \wedge y) \vee (\neg x \wedge \neg y)))$ é tautologia?".*

4.4 Método da Resolução

Em 1965, o matemático americano John Alan Robinson publicou um artigo descrevendo um método algorítmico que resolve SAT com duas características: é eficiente para algumas entradas e eficiente em reconhecê-las. A ideia básica consiste em que uma fórmula conjuntiva (composta por conjunções) é insatisfatível se alguma das suas subfórmulas for insatisfatível. Assim, o método só receberia como entrada fórmulas nessa condição.

4.4.1 Forma Normal Conjuntiva (FNC)

O método da resolução só aceita fórmulas como entrada que estejam em sua forma normal conjuntiva. Vamos abordar isso mais a fundo.

Definição 4.4.1 — Literal. Uma fórmula é um **literal** se for atômica ou a negação de uma atômica *(por exemplo, x, $\neg y$, z...)*.

Definição 4.4.2 — Cláusula. Uma fórmula é uma **cláusula** se for disjunção de literais *(por exemplo, $(A \vee B)$, $\neg y$, $(w \vee \neg x)$...)*.

Definição 4.4.3 — Forma Normal Conjuntiva. Uma fórmula está em sua **forma normal conjuntiva** se for conjunção de cláusulas *(por exemplo, $(A \vee B) \wedge (A \vee D)$, $\neg y$, $z \wedge (w \vee \neg x)$...)*.

Teorema 4.4.1 Toda proposição possui uma logicamente equivalente na FNC.

Demonstração. Queremos provar que, para toda $\varphi \in PROP$, existe $\varphi' \in PROP$ tal que φ' está na FNC e $\varphi' \equiv \varphi$. A prova será por indução.
Passo base. φ é atômica. Assim, $\varphi' = \varphi$.

Passo indutivo para φ da forma $\neg\psi$.
- Hipótese indutiva: $\exists\psi' \equiv \psi / \psi'$ está na FNC.
- Tese: $\exists(\neg\psi') \equiv \neg\psi / (\neg\psi')$ está na FNC.
Como $\psi' \equiv \psi$, $\neg\psi' \equiv \neg\psi$.
Além disso, podemos transformar $\neg\psi'$ para a FNC usando as equivalências lógicas: Lei da Dupla Negação, Leis de De Morgan e Distributividade.
Assim, temos $(\neg\psi') \equiv \neg\psi / (\neg\psi')$ está na FNC, concluindo a tese.

Passo indutivo para φ da forma $\rho\Box\theta$, onde $\Box \in \{\wedge, \vee, \rightarrow\}$.
- Hipótese indutiva: **(1)** $\exists\rho' \equiv \rho / \rho'$ está na FNC; **(2)** $\exists\theta' \equiv \theta / \theta'$ está na FNC.
- Tese: $\exists(\rho'\Box\theta') \equiv (\rho\Box\theta) / (\rho'\Box\theta')$ está na FNC.
Como $\rho' \equiv \rho$ e $\theta' \equiv \theta$, $(\rho'\Box\theta') \equiv (\rho\Box\theta)$.
Ainda, como ρ' e θ' estão na FNC,
$*$ para $\Box = \wedge$: $(\rho' \wedge \theta')$ está na FNC;
$*$ para $\Box = \vee$: podemos usar as equivalências lógicas para transformar $(\rho' \vee \theta')$ para a FNC;
$*$ para $\Box = \rightarrow$: podemos usar as equivalências lógicas para transformar $(\rho' \rightarrow \theta')$ para a FNC.
Assim, $(\rho'\Box\theta') \equiv (\rho\Box\theta) / (\rho'\Box\theta')$ está na FNC, concluindo a tese. ■

4.4.2 A regra da resolução

A resolução consiste em aplicar sucessivamente a seguinte equivalência lógica:

$$(x \vee \mathbf{y}) \wedge (\neg\mathbf{y} \vee z) \equiv (x \vee y) \wedge (\neg y \vee z) \wedge (x \vee z)$$

Ou seja, procuramos encontrar literais ditos **complementares** em cláusulas distintas e criar uma nova cláusula com os literais restantes. Se, ao fazermos esse processo, encontrarmos a **cláusula vazia** (sem literais), a fórmula é insatisfatível *(o método faz provas por contradição)*.

■ **Exemplo 4.6** $(\neg A \wedge E \wedge (\neg B \vee \neg C) \wedge (A \vee B) \wedge (\neg B \vee C \vee \neg D) \wedge (D \vee \neg E))$ é insatisfatível?

Suponha C_k para a k-ésima cláusula.
Tomando C_1 e C_4: (B) - C_7 *(uma nova conjunção foi feita com ela)*;
Tomando C_7 e C_3: $(\neg C)$ - C_8 *(atendendo à regra, podemos fazer a combinação* ***que quisermos****)*;
Tomando C_7 e C_5: $(C \vee \neg D)$ - C_9;
Tomando C_8 e C_9: $(\neg D)$ - C_{10};
Tomando C_{10} e C_6: $(\neg E)$ - C_{11};
Tomando C_{11} e C_2: $()$ *(a cláusula vazia)*;

Com o surgimento da cláusula vazia, uma contradição foi encontrada. Logo, a resposta é **sim**. Note que poderíamos ter gerado mais cláusulas (combinando, por exemplo, C_5 e C_6). ■

■ **Exemplo 4.7** Prove que $A \models ((A \rightarrow (B \vee C)) \rightarrow (((\neg C) \wedge D) \rightarrow B))$.

Aqui nos deparamos com dois problemas. Primeiro: o método não responde se φ é consequência lógica de Γ. Porém, temos que $\Gamma \models \varphi$ **se, e somente se,** $\Gamma \cup \{\neg\varphi\}$ **é insatisfatível**. Além disso, ele só aceita como entrada fórmulas na FNC. Para continuar, devemos transformar o conjunto.

$$\Gamma \cup \{\neg\varphi\} = \{A, \neg((A \rightarrow (B \vee C)) \rightarrow (((\neg C) \wedge D) \rightarrow B))\}$$

- (A): está na FNC *(olharemos cada elemento do conjunto)*;

- $\neg((A \rightarrow (B \vee C)) \rightarrow (((\neg C) \wedge D) \rightarrow B))$:
  $\equiv \neg((\neg A \vee (B \vee C)) \rightarrow (\neg((\neg C) \wedge D) \vee B))$ *("retirando"as implicações mais internas)*;
  $\equiv \neg((\neg A \vee B \vee C) \rightarrow ((C \vee \neg D) \vee B))$ *(aplicando as Leis de De Morgan)*;
  $\equiv \neg(\neg(\neg A \vee B \vee C) \vee (C \vee \neg D \vee B))$ *(retirando a implicação mais externa)*;
  $\equiv \neg((A \wedge \neg B \wedge \neg C) \vee (C \vee \neg D \vee B))$;
  $\equiv \neg(A \wedge \neg B \wedge \neg C) \wedge \neg(C \vee \neg D \vee B)$;
  $\equiv (\neg A \vee B \vee C) \wedge \neg C \wedge D \wedge \neg B$ *(Forma Normal Conjuntiva)*;

Assim, $\Gamma \cup \{\neg\varphi\} = \{A, (\neg A \vee B \vee C), \neg C, D, \neg B\}$.

Suponha C_k para a k-ésima cláusula.
Tomando C_1 e C_2: $(B \vee C)$ - C_6;
Tomando C_3 e C_6: (B) - C_7;
Tomando C_5 e C_7: $()$;

Com o surgimento da cláusula vazia, uma contradição foi encontrada. Logo, $\Gamma \cup \{\neg\varphi\}$ é insatisfatível e, portanto, $\Gamma \models \varphi$. ■

Você percebeu que a abordagem que fizemos para resolver o problema foi um pouco diferente nos dois exemplos? No 4.7, usamos notação de conjuntos, enquanto no 4.6, tratamos a fórmula em sua forma pura (a FNC). A questão é: computacionalmente, a abordagem por conjuntos é mais eficiente. Porém, nós, como seres inteligentes, **somos livres para escolher qualquer uma delas**.

4.4.3 Acelerando a resolução

Veremos adiante que a resolução também é custosa. Porém, com duas técnicas diferenciais (que a tornam especial), podemos diminuir significantemente seu custo.

Cláusulas de Horn

Nos anos 1950, o matemático também americano Alfred Horn estudou intensamente as propriedades de um determinado tipo da lógica proposicional que tinham um certo apelo à representação de "condições implicam numa consequência":

$$(x_1 \wedge x_2 \wedge x_3 \wedge \ldots \wedge x_n) \rightarrow y$$
$$\textit{onde } y \textit{ e todos os } x_i \textit{ são proposições atômicas (literais positivos)}$$
$$\equiv \neg(x_1 \wedge x_2 \wedge x_3 \wedge \ldots \wedge x_n) \vee y$$
$$\equiv (\neg x_1 \vee \neg x_2 \vee \neg x_3 \vee \ldots \vee \neg x_n \vee y)$$

A essas cláusulas especiais foram dadas o nome de cláusulas de Horn.

Definição 4.4.4 — Cláusula de Horn. Uma cláusula é dita **cláusula de Horn** se, e somente se, contém, no máximo, um literal positivo.

Quando todas as cláusulas de uma fórmula são cláusulas de Horn, o esforço para procurar literais complementares em toda a cláusula é poupado: precisamos apenas olhar os literais positivos (que será no máximo um por cláusula).

Propagação da Cláusula Unitária

Quando uma cláusula contém apenas um literal, a designamos **cláusula unitária**. Elas são especiais porque é graças a elas que podemos encontrar a cláusula vazia:

Lema 4.4.2 Seja uma fórmula $\varphi \in PROP$ na FNC. Se φ não tem cláusulas unitárias, então φ é satisfatível.

Demonstração. Esse lema é muito poderoso para a computação! A prova será por contradição.

(1): Seja φ uma proposição na FNC sem cláusulas unitárias e insatisfatível.
(2): Por **1**, existe uma contradição entre dois conjuntos de cláusulas C_1 e C_2 de φ $(C_1 \wedge C_2 = 0)$.
(3): Por **2**, temos que $C_2 \equiv \neg C_1$.
(4): Pela Lei de De Morgan, podemos escrever C_2 como conjunção das negações dos literais de C_1. Assim, cada literal formará uma cláusula unitária.

Por **1** e **4**, temos uma contradição, pois nossa fórmula não tem cláusulas unitárias.
A prova está completa. ■

Assim, ao fazermos a regra da resolução, damos prioridade para as cláusulas unitárias (considerando que novas cláusulas unitárias podem surgir). Isso nos poupa o trabalho de aplicar a regra em todas as cláusulas possíveis. *Cuidado! A recíproca não é verdadeira. Por exemplo:* $(\neg A \vee A) \wedge B$.

4.4.4 Complexidade computacional

No pior caso, o custo do método da resolução é, também, exponencial — $O(2^n)$. Contudo, as duas maneiras de acelerá-lo tornam-no muito amigável: se uma entrada só contém cláusulas de Horn, precisamos visitar apenas os literais positivos em todas as cláusulas, ou seja, o número de operações é $k \times n$, onde k é uma constante e n o número de cláusulas — o custo do método torna-se $O(n)$, ou seja, linear! É por isso que o método da Resolução é o mais atraente: linguagens de programação como Prolog — usada amplamente em inteligência artificial —, por exemplo, se baseiam nele.

4.5 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. Daí, se propôs a 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 valoração-verdade, seu método partiu da noção de **regra de dedução**, que independe de uma interpretação em valores booleanos. Daí, para cada operador lógico, Gentzen definiu dois conjuntos de regras de dedução.

4.5.1 As regras de dedução

Suponha que ϕ, uma fórmula qualquer, seja verdade. Se, a partir dela, conseguimos deduzir ψ, podemos deduzir seguramente $(\phi \rightarrow \psi)$. Isso é um exemplo do que vem a seguir.

Regras de Introdução determinam as **condições mínimas** para que se pudesse deduzir uma proposição cujo operador principal fosse o em questão;

Para deduzirmos	Precisamos de
$\dfrac{}{(\phi \wedge \psi)}$	$\dfrac{\phi \quad \psi}{(\phi \wedge \psi)}$
$\dfrac{}{(\phi \vee \psi)}$	$\dfrac{\phi}{(\phi \vee \psi)}$ $\quad$ $\dfrac{\psi}{(\phi \vee \psi)}$
$\dfrac{}{(\phi \rightarrow \psi)}$	$\dfrac{\begin{array}{c}[\phi] \\ \vdots \\ \psi\end{array}}{(\phi \rightarrow \psi)}$

$[\phi]$ significa que ϕ é uma **suposição** ou **premissa**.

Regras de Eliminação determinam as **consequências imediatas** que poderiam ser obtidas a partir de uma proposição cujo operador principal fosse o em questão.

Se temos	Podemos deduzir
$\dfrac{(\phi \wedge \psi)}{}$	$\dfrac{(\phi \wedge \psi)}{\phi}$ $\quad$ $\dfrac{(\phi \wedge \psi)}{\psi}$
$\dfrac{(\phi \vee \psi)}{}$	$\dfrac{\begin{array}{cc}(\phi \vee \psi) & \\ [\phi] & [\psi] \\ \vdots & \vdots \\ \theta & \theta\end{array}}{\theta}$
$\dfrac{\phi \quad (\phi \rightarrow \psi)}{}$	$\dfrac{\phi \quad (\phi \rightarrow \psi)}{\psi}$

4.5.2 Dedutibilidade

A pergunta que o método procura responder é *"ϕ é dedutível?"*, em contrapartida às perguntas usuais dos outros métodos.

Definição 4.5.1 — Dedutibilidade. Dizemos que uma sentença φ é dedutível, a partir de um conjunto de sentenças Γ, se existe uma árvore de dedução cuja raiz é φ, cujos vértices são aplicações das regras de dedução e cujas folhas são sentenças de Γ ou suposições adicionais devidamente descartadas. A notação é $\Gamma \vdash \varphi$.

Pode-se estabelecer uma relação entre consequência lógica e dedutibilidade, apesar de serem conceitos distintos: $\Gamma \vdash \varphi \leftrightarrow \Gamma \models \varphi$.

■ **Exemplo 4.8** $\{(\varphi \to \psi), (\varphi \to \theta)\} \vdash (\varphi \to \theta)$?

Inicialmente, nós montamos a árvore *de baixo para cima*, aplicando as regras, para preencher nosso conjunto de premissas.

Iniciamos com:
$$\frac{}{(\varphi \to \theta)}$$

Pela regra $\to$-introdução,
$$\frac{\begin{array}{c}[\varphi]^1 \\ \theta\end{array}}{(\varphi \to \theta)}$$

Aqui é nosso limite. Assim, nossas premissas são $\{[\varphi]\}$. Agora, *de cima para baixo*, vamos usar as sentenças do conjunto, as premissas e as regras para tentar **deduzir a raiz**.

Pela regra $\to$-eliminação,
$$\frac{[\varphi]^1 \quad (\varphi \to \psi)}{\psi}$$

Pela regra $\to$-eliminação,
$$\frac{\dfrac{[\varphi]^1 \quad (\varphi \to \psi)}{\psi} \qquad (\psi \to \theta)}{\theta}$$

Pela regra $\to$-introdução,
$$\frac{\dfrac{\dfrac{[\varphi]^1 \quad (\varphi \to \psi)}{\psi} \qquad (\psi \to \theta)}{\theta}}{(\varphi \to \theta)} \; //^1$$

Regras que utilizam de premissas, ao serem aplicadas no processo de cima para baixo, provocam **descarte de premissas** *(foi o caso com o $[\varphi]$)*. Não podemos reutilizar uma premissa descartada e precisamos garantir que todas sejam devidamente descartadas.

Assim, temos que tal árvore de dedução existe. Logo, a resposta do método é **sim**. ■

■ **Exemplo 4.9** $((\varphi \to \psi) \to ((\theta \wedge \varphi) \to (\theta \wedge \psi)))$ é tautologia?

O método não responde se uma proposição ϕ é tautologia. Contudo, temos que ϕ **é tautologia se, e somente se,** $\emptyset \models \phi$ (ou seja, $\neg\phi$ é insatisfatível). Assim, podemos avançar.

Iniciamos com:
$$\frac{}{((\varphi \to \psi) \to ((\theta \wedge \varphi) \to (\theta \wedge \psi)))}$$

$$[\varphi \to \psi]^1$$

Pela regra →-introdução,
$$\frac{((\theta \wedge \varphi) \to (\theta \wedge \psi))}{((\varphi \to \psi) \to ((\theta \wedge \varphi) \to (\theta \wedge \psi)))}$$

$$[\varphi \to \psi]^1 [\theta \wedge \varphi]^2$$

Pela regra →-introdução,
$$\frac{\dfrac{(\theta \wedge \psi)}{((\theta \wedge \varphi) \to (\theta \wedge \psi))}}{((\varphi \to \psi) \to ((\theta \wedge \varphi) \to (\theta \wedge \psi)))}$$

$$[\varphi \to \psi]^1 [\theta \wedge \varphi]^2$$

Pela regra ∧-introdução,
$$\frac{\dfrac{\dfrac{\theta \quad \psi}{(\theta \wedge \psi)}}{((\theta \wedge \varphi) \to (\theta \wedge \psi))}}{((\varphi \to \psi) \to ((\theta \wedge \varphi) \to (\theta \wedge \psi)))}$$

Não conseguimos mais avançar. Assim, nosso conjunto de premissas é $\{[\varphi \to \psi], [\theta \wedge \varphi]\}$. Tentemos deduzir a raiz com elas:

Pela regra ∧-eliminação,
$$\frac{[\theta \wedge \varphi]^2}{\theta} \qquad \frac{[\theta \wedge \varphi]^2}{\varphi}$$

Pela regra →-eliminação,
$$\frac{[\theta \wedge \varphi]^2}{\theta} \qquad \frac{\dfrac{[\theta \wedge \varphi]^2}{\varphi} \qquad [\varphi \to \psi]^1}{\psi}$$

Pela regra ∧-introdução,
$$\frac{\dfrac{[\theta \wedge \varphi]^2}{\theta} \qquad \dfrac{\dfrac{[\theta \wedge \varphi]^2}{\varphi} \qquad [\varphi \to \psi]^1}{\psi}}{(\theta \wedge \psi)}$$

Pela regra →-introdução,
$$\frac{\dfrac{\dfrac{[\theta \wedge \varphi]^2}{\theta} \qquad \dfrac{\dfrac{[\theta \wedge \varphi]^2}{\varphi} \qquad [\varphi \to \psi]^1}{\psi}}{(\theta \wedge \psi)}}{((\theta \wedge \varphi) \to (\theta \wedge \psi))} \; //^2$$

Pela regra →-introdução,
$$\frac{\dfrac{\dfrac{\dfrac{[\theta \wedge \varphi]^2}{\theta} \qquad \dfrac{\dfrac{[\theta \wedge \varphi]^2}{\varphi} \qquad [\varphi \to \psi]^1}{\psi}}{(\theta \wedge \psi)}}{((\theta \wedge \varphi) \to (\theta \wedge \psi))} \; //^2}{((\varphi \to \psi) \to ((\theta \wedge \varphi) \to (\theta \wedge \psi)))} \; //^1$$

Concluímos que tal árvore de prova existe. Assim, a resposta do método é **sim**. ∎

4.5.3 A negação

Note que não mencionamos a negação no conjunto de regras de dedução nem nos exemplos. Isso porque a negação não tem uma regra própria. Além do mais, o método da dedução natural, assim como a própria lógica e seus métodos de prova, sofreram "adaptações"ao longo do tempo para lidar com ela.

Inicialmente, temos que:

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

Assim, usamos as regras da implicação para lidar com a negação.

■ **Exemplo 4.10** $((\neg\varphi \vee \psi) \to (\neg\psi \to \neg\varphi))$ é tautologia?

Onde encontrarmos a negação, faremos uma substituição para a implicação equivalente.

Iniciamos com:
$$\frac{}{(((\varphi \to \bot) \vee \psi) \to ((\psi \to \bot) \to (\varphi \to \bot)))}$$

$$[(\varphi \to \bot) \vee \psi]^1$$

Pela regra →-introdução,
$$\frac{((\psi \to \bot) \to (\varphi \to \bot))}{(((\varphi \to \bot) \vee \psi) \to ((\psi \to \bot) \to (\varphi \to \bot)))}$$

$$[(\varphi \to \bot) \vee \psi]^1 [\psi \to \bot]^2$$

Pela regra →-introdução,
$$\frac{\dfrac{(\varphi \to \bot)}{((\psi \to \bot) \to (\varphi \to \bot))}}{(((\varphi \to \bot) \vee \psi) \to ((\psi \to \bot) \to (\varphi \to \bot)))}$$

$$[(\varphi \to \bot) \vee \psi]^1 [\psi \to \bot]^2 [\varphi]^3$$

Pela regra →-introdução,
$$\frac{\dfrac{\dfrac{\bot}{(\varphi \to \bot)}}{((\psi \to \bot) \to (\varphi \to \bot))}}{(((\varphi \to \bot) \vee \psi) \to ((\psi \to \bot) \to (\varphi \to \bot)))}$$

Nossas premissas são $\{[(\varphi \to \bot) \vee \psi], [\psi \to \bot], [\varphi]\}$. Vamos tentar deduzir a raiz.

Pela regra ∨-eliminação,
$$\frac{[(\varphi \to \bot) \vee \psi]^1}{[\varphi \to \bot]^4 \quad [\psi]^5}$$

∗ *Atenção! A regra nos diz que, a partir desse momento, devemos ser capazes de inferir uma mesma sentença de* $[\varphi \to \bot]$ *e* $[\psi]$*. Se conseguirmos, podemos afirmá-la.*

Pela regra →-eliminação,
$$\frac{[(\varphi \to \bot) \vee \psi]^1}{\dfrac{[\varphi]^3 [\varphi \to \bot]^4}{\bot} \quad \dfrac{[\psi]^5 [\psi \to \bot]}{\bot}}$$

∗ *Encontramos* $\bot$ *a partir de ambas as premissas: podemos deduzi-lo seguramente. Além disso, como o ∨-eliminação é uma regra que usa premissas e está sendo usada para a dedução da raiz (de cima para baixo), haverá um descarte.*

Pela regra ∨-eliminação (término),

$$\dfrac{\dfrac{[(\varphi \to \bot) \vee \psi]^1}{\dfrac{[\varphi]^3[\varphi \to \bot]^4}{\bot} \quad \dfrac{[\psi]^5[\psi \to \bot]^2}{\bot}}}{\bot} \; //^4 //^5$$

Pela regra →-introdução,

$$\dfrac{\dfrac{\dfrac{[(\varphi \to \bot) \vee \psi]^1}{\dfrac{[\varphi]^3[\varphi \to \bot]^4}{\bot} \quad \dfrac{[\psi]^5[\psi \to \bot]^2}{\bot}}}{\bot}}{(\varphi \to \bot)} \; //^4 //^5 \; //^3$$

Pela regra →-introdução,

$$\dfrac{\dfrac{\dfrac{\dfrac{[(\varphi \to \bot) \vee \psi]^1}{\dfrac{[\varphi]^3[\varphi \to \bot]^4}{\bot} \quad \dfrac{[\psi]^5[\psi \to \bot]^2}{\bot}}}{\bot}}{(\varphi \to \bot)}}{(\psi \to \bot) \to (\varphi \to \bot)} \; //^4 //^5 \; //^3 \; //^2$$

Pela regra →-introdução,

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[(\varphi \to \bot) \vee \psi]^1}{\dfrac{[\varphi]^3[\varphi \to \bot]^4}{\bot} \quad \dfrac{[\psi]^5[\psi \to \bot]^2}{\bot}}}{\bot}}{(\varphi \to \bot)}}{(\psi \to \bot) \to (\varphi \to \bot)}}{(((\varphi \to \bot) \vee \psi) \to ((\psi \to \bot) \to (\varphi \to \bot)))} \; //^4 //^5 \; //^3 \; //^2 \; //^1$$

Temos que a árvore de dedução existe. Logo, a resposta do método é **sim**. ∎

4.5.4 As três lógicas

Como dissemos anteriormente, a lógica teve de se adaptar para incluir a negação em seus métodos dedutivos. Um dos motivos será mostrado no exemplo abaixo.

∎ **Exemplo 4.11** $(\neg\phi \to (\phi \to \psi))$ é tautologia?

De baixo para cima, obtemos o seguinte conjunto de premissas: $\{[\phi \to \bot], [\phi]\}$. Veja o que acontece ao tentarmos deduzir a raiz.

Pela regra →-eliminação, $\dfrac{[\phi][\phi \to \bot]}{\bot}$

Não podemos mais avançar. Como não conseguimos montar a árvore de dedução — não deduzimos a raiz nem fizemos o descarte devido das premissas —, a resposta do método é **não**. Contudo:

ϕ	ψ	$\neg\phi$	$(\phi \to \psi)$	$(\neg\phi \to (\phi \to \psi))$
0	0	1	1	1
0	1	1	1	1
1	0	0	0	1
1	1	0	1	1

A tabela-verdade mostra o contrário. ∎

Precisamos fazer com que o método atinja sua corretude. Para isso, é necessário a introdução de mais uma regra:

Se temos	Podemos deduzir
$\dfrac{\bot}{}$	$\dfrac{\bot}{\varphi}$, *se* φ *é atômica.*

A regra é chamada **Princípio da Explosão** ou *Ex Falsum Quolibet — a partir do falso, deduzimos qualquer coisa*. Podemos usá-la para deduzir ψ no exemplo anterior.

Sua adição ajuda bastante o método da Dedução Natural, mas, infelizmente, ainda não é suficiente. Veja os dois exemplos abaixo.

■ **Exemplo 4.12** $(\neg\phi \vee \phi)$ é tautologia?

Vamos buscar as premissas.

Iniciamos com: $\dfrac{}{((\phi \to \bot) \vee \phi)}$

Pela regra $\vee$*-introdução,* $\dfrac{(\phi \to \bot)}{((\phi \to \bot) \vee \phi)} \quad \dfrac{\phi}{((\phi \to \bot) \vee \phi)}$

Pela regra $\to$*-introdução,* $\dfrac{\dfrac{\bot}{(\phi \to \bot)}}{((\phi \to \bot) \vee \phi)} \quad \dfrac{[\phi]^1}{((\phi \to \bot) \vee \phi)}$

Temos $[\phi]$ como única premissa. Ao tentarmos deduzir a raiz, note que se quisermos deduzir direto a expressão usando $\vee$*-introdução* (nossa única opção), não faremos um descarte devido de $[\phi]$. Assim, o método responderá erroneamente com um **não**. ■

■ **Exemplo 4.13** $\neg\neg\phi \models \phi$?

Iniciamos com: $\dfrac{}{\phi}$

Note que $((\phi \to \bot) \to \bot)$ está disponível, mas não podemos aplicar nenhuma regra. Assim, o método encerrará aqui com um **não** também errado. ■

Novamente, precisamos adicionar mais uma regra dedutiva (que nós já conhecemos!).

Para deduzirmos	Precisamos de
$\dfrac{}{\phi}$	$\dfrac{\begin{array}{c}[\neg\phi] \\ \cdots \\ \bot\end{array}}{\phi}$

Essa regra se chama **Redução ao Absurdo** ou *Reductio ad Absurdum — se, ao supormos o contrário de uma sentença, chegamos a uma contradição, podemos deduzir a sentença*. Podemos supor $[\neg\phi]$ em ambos os exemplos e aplicar as regras usuais para concluir a sentença. Para reforçar: ao usar a regra na montagem de cima para baixo, um descarte na suposição será provocado.

As sentenças dos dois exemplos anteriores não foram escolhidas ao acaso: elas são duas **leis do pensamento**:

Lei do Terceiro Excluído *Tertium nom Datur* Uma proposição é ou verdadeira, ou falsa. — $(\neg\phi \vee \phi)$ *é tautologia.*

Lei da Dupla Negação *Duplex Negatio Affirmat* Se uma proposição é verdadeira, não é o caso de que não seja verdadeira. — $\neg\neg\phi \equiv \phi$

Definição 4.5.2 — As Três Lógicas. As duas leis do pensamento, junto com o Princípio da Explosão, dividem a lógica em três vertentes.

Lógica de Relevância (ou Minimal) não admite nenhuma das leis em suas demonstrações;

Lógica Intuicionista admite o Princípio da Explosão, mas não admite o restante, em suas demonstrações;

Lógica Clássica admite a Lei do Terceiro Excluído, a Lei da Dupla Negação — a Redução ao Absurdo — e o Princípio da Explosão em suas demonstrações;

De cima para baixo, cada uma delas é uma versão mais forte que a anterior.

4.5.5 Forma normal e redundâncias

Uma outra característica positiva do método da Dedução Natural é a capacidade de avaliar **matematicamente** diversas soluções para o mesmo problema e definir **a melhor** (não conseguimos fazer isso com o método dos Tableaux Analíticos nem com o da Resolução): uma árvore de prova é dita estar na **forma normal** — a melhor solução — se não contém redundâncias.

Definição 4.5.3 — Redundância. Quando aplicamos as regras de introdução e eliminação em sequência **para o mesmo operador**, temos uma **redundância**. Podem ser divididas em 2 tipos:

- Aplicar uma regra de introdução seguida de uma de eliminação (**tipo** β);
- Aplicar uma regra de eliminação seguida de uma de introdução (**tipo** η);

Normalização

Mesmo quando uma árvore de prova não está na forma normal, ela sempre pode ser **normalizada**, com um conjunto de regras chamadas **regras de redução**, descritas abaixo. Esse fato foi provado em 1965 pelo lógico sueco Dag Prawitz.

Teorema 4.5.1 — Teorema da Normalização. Toda árvore de prova em Dedução Natural que contenha redundâncias pode ser normalizada.

Regras de Redução

Existe um conjunto de 6 regras, que nos permite normalizar uma árvore de prova com redundâncias, dividido de acordo com os operadores e os tipos de redundâncias.

- Regras de redução para redundâncias β *(*Δ *e* ∇ *representam subárvores)*:

Operador	Redundância			Redução		
$\wedge$	$\dfrac{\dfrac{\nabla_1 \quad \nabla_2}{\varphi \quad \psi}}{\dfrac{(\varphi \wedge \psi)}{\varphi}}$ Δ_3	ou	$\dfrac{\dfrac{\nabla_1 \quad \nabla_2}{\varphi \quad \psi}}{\dfrac{(\varphi \wedge \psi)}{\psi}}$ Δ_4	∇_1 φ Δ_3	ou	∇_2 ψ Δ_4
$\vee$	$\dfrac{\dfrac{\nabla_1}{\varphi}}{\dfrac{(\varphi \vee \psi)}{\dfrac{[\varphi] \quad [\psi]}{\dfrac{\Delta_3 \quad \Delta_4}{\dfrac{\theta \quad \theta}{\theta}}}}}$ Δ_5	ou	$\dfrac{\dfrac{\nabla_2}{\psi}}{\dfrac{(\varphi \vee \psi)}{\dfrac{[\varphi] \quad [\psi]}{\dfrac{\Delta_3 \quad \Delta_4}{\dfrac{\theta \quad \theta}{\theta}}}}}$ Δ_5	∇_1 φ Δ_3 θ Δ_5	ou	∇_2 ψ Δ_4 θ Δ_5
$\rightarrow$	$\dfrac{\nabla_2 \quad \dfrac{\dfrac{[\varphi]}{\Delta_1}}{\psi}}{\dfrac{\varphi \quad (\varphi \rightarrow \psi)}{\psi}}$ Δ_3					∇_2 φ Δ_1 ψ Δ_3

- Regras de redução para redundâncias η:

Operador	Redundância	Redução
$\wedge$	$\dfrac{\dfrac{\nabla_1 \quad \nabla_1}{(\varphi \wedge \psi) \quad (\varphi \wedge \psi)}}{\dfrac{\varphi \quad \psi}{(\varphi \wedge \psi)}}$ Δ_2	∇_1 $(\varphi \wedge \psi)$ Δ_2
$\vee$	$\dfrac{\dfrac{\nabla_1}{(\varphi \vee \psi)}}{\dfrac{[\varphi] \quad [\psi]}{\dfrac{(\varphi \vee \psi) \quad (\varphi \vee \psi)}{(\varphi \vee \psi)}}}$ Δ_2	∇_1 $(\varphi \vee \psi)$ Δ_2
$\rightarrow$	$\dfrac{[\varphi] \quad \nabla_1}{\dfrac{(\varphi \rightarrow \psi)}{\psi}}{(\varphi \rightarrow \psi)}$ Δ_2	∇_1 $(\varphi \rightarrow \psi)$ Δ_2

Podemos medir o tamanho da fórmula no centro da redundância: para as do tipo β, ela é dita **máxima**. Para as do tipo η, ela é dita **mínima**. Isso nos permite determinar quão longe uma árvore de dedução está de sua forma normal.

4.6 Método do Cálculo de Sequentes

Também definido por Gentzen, o método do cálculo de sequentes surgiu quando foi notado que as regras de dedução não são inversíveis — só podem ser aplicadas em um único sentido por vez. O método é semelhante ao da Dedução Natural, mas com regras inversíveis e dando à negação um tratamento condizente com os outros operadores. Ele toma, como base, **sequentes**.

Definição 4.6.1 — Sequente. Um sequente é uma estrutura do tipo:

$$\phi_1, \phi_2, \phi_3, ..., \phi_n \vdash \psi_1, \psi_2, \psi_3, ..., \psi_n$$

Onde $\phi_1, \phi_2, \phi_3, ..., \phi_n$ são as **premissas** do sequente e $\psi_1, \psi_2, \psi_3, ..., \psi_n$ são as **conclusões** do sequente, de modo que **se** $(\phi_1 \wedge \phi_2 \wedge \phi_3 \wedge ... \wedge \phi_n)$**, então** $(\psi_1 \vee \psi_2 \vee \psi_3 \vee ... \vee \psi_n)$.

4.6.1 As regras de dedução

O método utiliza dois tipos de regras simples de dedução para cada operador lógico *(Γ e Δ são conjuntos de fórmulas).*

Introdução à Direita Regras que podemos aplicar a uma fórmula à direita de $\vdash$.

Para deduzirmos	Precisamos de
$\dfrac{}{\Gamma \vdash (\neg\phi), \Delta}$	$\dfrac{\Gamma, \phi \vdash \Delta}{\Gamma \vdash (\neg\phi), \Delta}$
$\dfrac{}{\Gamma \vdash (\phi \wedge \psi), \Delta}$	$\dfrac{\Gamma \vdash \phi, \Delta}{\Gamma \vdash (\phi \wedge \psi), \Delta}$ $\quad$ $\dfrac{\Gamma \vdash \psi, \Delta}{\Gamma \vdash (\phi \wedge \psi), \Delta}$
$\dfrac{}{\Gamma \vdash (\phi \vee \psi), \Delta}$	$\dfrac{\Gamma \vdash \phi, \psi, \Delta}{\Gamma \vdash (\phi \vee \psi), \Delta}$
$\dfrac{}{\Gamma \vdash (\phi \to \psi), \Delta}$	$\dfrac{\Gamma, \phi \vdash \psi, \Delta}{\Gamma \vdash (\phi \to \psi), \Delta}$

Introdução à Esquerda Regras que podemos aplicar a uma fórmula à esquerda de $\vdash$.

Para deduzirmos	Precisamos de
$\dfrac{}{\Gamma, (\neg\phi) \vdash \Delta}$	$\dfrac{\Gamma \vdash \phi, \Delta}{\Gamma, (\neg\phi) \vdash \Delta}$
$\dfrac{}{\Gamma, (\phi \wedge \psi) \vdash \Delta}$	$\dfrac{\Gamma, \phi, \psi \vdash \Delta}{\Gamma, (\phi \wedge \psi) \vdash \Delta}$
$\dfrac{}{\Gamma, (\phi \vee \psi) \vdash \Delta}$	$\dfrac{\Gamma, \phi \vdash \Delta}{\Gamma, (\phi \vee \psi) \vdash \Delta}$ $\quad$ $\dfrac{\Gamma, \psi \vdash \Delta}{\Gamma, (\phi \vee \psi) \vdash \Delta}$
$\dfrac{}{\Gamma, (\phi \to \psi) \vdash \Delta}$	$\dfrac{\Gamma_1, \psi \vdash \Delta_1 \quad \Gamma_2 \vdash \phi, \Delta_2}{\Gamma_1, \Gamma_2, (\phi \to \psi) \vdash \Delta_1, \Delta_2}$

As regras de dedução só podem ser aplicadas em fórmulas **adjacentes ao** $\vdash$. Para permitir isso (e auxiliar na prova), temos mais regras que permitem estruturar o sequente — **regras estruturais**, que podem ser aplicadas a qualquer momento, tanto à esquerda quanto à direita:

Se temos	Podemos deduzir	
$\dfrac{\Gamma, \phi, \psi \vdash \Delta}{}$	$\dfrac{\Gamma, \phi, \psi \vdash \Delta}{\Gamma, \psi, \phi \vdash \Delta}$	**Permutação**
$\dfrac{\Gamma, \phi, \phi \vdash \Delta}{}$	$\dfrac{\Gamma, \phi, \phi \vdash \Delta}{\Gamma, \phi \vdash \Delta}$	**Contração**
$\dfrac{\Gamma \vdash \Delta}{}$	$\dfrac{\Gamma \vdash \Delta}{\Gamma, \phi \vdash \Delta}$	**Enfraquecimento**

Temos mais uma regra especial:

$\dfrac{\Gamma_1 \vdash \phi, \Delta_1 \quad \Gamma_2, \phi \vdash \Delta_2}{}$	$\dfrac{\Gamma_1 \vdash \phi, \Delta_1 \quad \Gamma_2, \phi \vdash \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}$	**Regra do Corte**

Finalmente, temos a condição de parada do método:

$$\phi \vdash \phi \quad \textbf{Axioma}$$

Quando todas as folhas da árvore de dedução forem axiomas, o método encerra, respondendo "sim".

■ **Exemplo 4.14** $\{(x \to y), (y \to z)\} \vdash (x \to (w \vee z))$?

Ao contrário da Dedução Natural, possuímos um único sentido de montar a árvore — de baixo para cima —, pois o sequente já possui premissas em sua definição. As sentenças do conjunto estarão do lado esquerdo (as premissas), enquanto a de fora estará do lado direito (a conclusão).

Note também que o método se assemelha muito ao dos Tableaux Analíticos: podemos tomar o lado esquerdo do sequente como o lado do **verdadeiro** ($\hat{v}(\phi) = 1$) e o lado direito como o lado do **falso** ($\hat{v}(\psi) = 0$). Além disso, não continuamos em um ramo onde um axioma foi encontrado, similar à contradição no tableau.

Iniciamos com:

$$\frac{}{(x \to y), (y \to z) \vdash (x \to (w \vee z))}$$

$\to$*-Introdução à direita:*

$$\frac{(x \to y), (y \to z), x \vdash (w \vee z)}{(x \to y), (y \to z) \vdash (x \to (w \vee z))}$$

$\vee$*-Introdução à direita:*

$$\frac{\dfrac{(x \to y), (y \to z), x \vdash w, z}{(x \to y), (y \to z), x \vdash (w \vee z)}}{(x \to y), (y \to z) \vdash (x \to (w \vee z))}$$

Permutação:

$$\frac{\dfrac{\dfrac{(x \to y), x, (y \to z) \vdash w, z}{(x \to y), (y \to z), x \vdash w, z}}{(x \to y), (y \to z), x \vdash (w \vee z)}}{(x \to y), (y \to z) \vdash (x \to (w \vee z))}$$

$*$ *Nesse momento, a próxima regra nos diz para tomar um conjunto de sentenças do lado esquerdo junto com o consequente e um outro do lado direito junto com o antecendente e separá-los.*

$\to$-Introdução à esquerda:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(x\to y),x\vdash y,w \quad \mathbf{z\vdash z}}{(x\to y),x,(y\to z)\vdash w,z}}{(x\to y),(y\to z),x\vdash w,z}}{(x\to y),(y\to z),x\vdash (w\vee z)}}{(x\to y),(y\to z)\vdash (x\to (w\vee z))}$$

Permutação:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x,(x\to y)\vdash y,w}{(x\to y),x\vdash y,w} \quad \mathbf{z\vdash z}}{(x\to y),x,(y\to z)\vdash w,z}}{(x\to y),(y\to z),x\vdash w,z}}{(x\to y),(y\to z),x\vdash (w\vee z)}}{(x\to y),(y\to z)\vdash (x\to (w\vee z))}$$

$\to$-Introdução à esquerda:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathbf{x\vdash x} \quad y\vdash y,w}{x,(x\to y)\vdash y,w}}{(x\to y),x\vdash y,w} \quad \mathbf{z\vdash z}}{(x\to y),x,(y\to z)\vdash w,z}}{(x\to y),(y\to z),x\vdash w,z}}{(x\to y),(y\to z),x\vdash (w\vee z)}}{(x\to y),(y\to z)\vdash (x\to (w\vee z))}$$

Enfraquecimento:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathbf{x\vdash x} \quad \dfrac{\mathbf{y\vdash y}}{y\vdash y,w}}{x,(x\to y)\vdash w}}{(x\to y),x\vdash y,w} \quad \mathbf{z\vdash z}}{(x\to y),x,(y\to z)\vdash w,z}}{(x\to y),(y\to z),x\vdash w,z}}{(x\to y),(y\to z),x\vdash (w\vee z)}}{(x\to y),(y\to z)\vdash (x\to (w\vee z))}$$

Todas as folhas são axiomas: a árvore de dedução existe e a resposta é **sim**. ∎

■ **Exemplo 4.15** $(\neg(A\vee B)\to(\neg A\wedge\neg B))$ é tautologia?

Vamos mostrar a árvore pronta, desta vez.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\text{(ENF)}\ \mathbf{A\vdash A}}{\text{(PER)}\ A\vdash B,A}}{\text{(¬-D)}\ A\vdash A,B}}{\text{(∧-D)}\vdash \neg A,A,B} \quad \dfrac{\mathbf{B\vdash B}\ \text{(ENF)}}{B\vdash A,B\ \text{(¬-D)}}}{\vdash \neg B,A,B\ \text{(∧-D)}}}{\vdash (\neg A\wedge\neg B),A,B\ \text{(PER)}}}{\vdash A,(\neg A\wedge\neg B),B\ \text{(PER)}}}{\vdash A,B,(\neg A\wedge\neg B)\ \text{(∨-E)}}}{\neg(A\vee B)\vdash(\neg A\wedge\neg B)\ \text{(¬-D)}}}{\vdash(\neg(A\vee B)\to(\neg A\wedge\neg B))}$$

A resposta é **sim**. ∎

■ **Exemplo 4.16** $((\neg A\to B)\wedge((\neg B\wedge(A\vee C))\wedge\neg C))$ é satisfatível?

Este último exemplo nos remete à pergunta original do Problema da Satisfatibilidade. O método do Cálculo de Sequentes não a responde, mas sabemos que uma proposição ϕ **é satisfatível** $\leftrightarrow \neg\phi$ **não é tautologia** $(\nvdash \neg\phi)$. Assim, avançamos.

$$\dfrac{\dfrac{\textbf{(→-E)}\ \mathbf{B} \vdash \mathbf{B} \qquad \dfrac{\dfrac{A \vdash C \ \textbf{(ENF)}}{A, A \vdash C \ \textbf{(¬-D)}}}{A \vdash \neg A, C \ \textbf{(→-E)}}}{\dfrac{\textbf{(PER)}\ A, (\neg A \to B) \vdash B, C}{\dfrac{\textbf{(¬-E)}\ (\neg A \to B), A \vdash B, C}{\dfrac{\textbf{(PER)}\ (\neg A \to B), A, \neg B \vdash C}{\textbf{(∨-E)}\ (\neg A \to B), \neg B, A \vdash C}}}} \qquad \dfrac{\dfrac{\textbf{(→-E)}\ \mathbf{B} \vdash \mathbf{B} \qquad \dfrac{\mathbf{C} \vdash \mathbf{C} \ \textbf{(ENF)}}{C \vdash \neg A, C \ \textbf{(→-E)}}}{\dfrac{C, (\neg A \to B) \vdash B, C \ \textbf{(PER)}}{\dfrac{(\neg A \to B), C \vdash B, C \ \textbf{(¬-E)}}{\dfrac{(\neg A \to B), C, \neg B \vdash C \ \textbf{(PER)}}{(\neg A \to B), \neg B, C \vdash C \ \textbf{(∨-E)}}}}}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(\neg A \to B), \neg B, (A \lor C) \vdash C \ \textbf{(∧-E)}}{(\neg A \to B), (\neg B \land (A \lor C) \vdash C \ \textbf{(¬-E)}}}{(\neg A \to B), (\neg B \land (A \lor C), \neg C \vdash \ \textbf{(∨-E)}}}{(\neg A \to B), ((\neg B \land (A \lor C)) \land \neg C) \vdash \ \textbf{(∨-E)}}}{((\neg A \to B) \land ((\neg B \land (A \lor C)) \land \neg C)) \vdash \ \textbf{(¬-D)}}}{\vdash \neg((\neg A \to B) \land ((\neg B \land (A \lor C)) \land \neg C))}$$

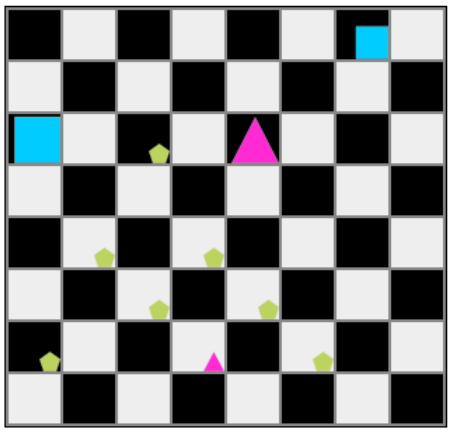
No vértice $A \vdash C$, nenhuma regra estrutural nos ajudará, assim, não encontraremos um axioma. Além disso, pela definição de sequente, $A \to C$ não é sempre verdade. A árvore de dedução não existe e o método responde com **não**. Assim, a fórmula é satisfatível. ∎

II Lógica de Primeira Ordem

5. Estruturas

5.1 Introdução à Primeira Ordem

Enquanto a lógica proposicional lida com expressões declarativas simples, a lógica de primeira ordem introduz uma linguagem mais rica, envolvendo **predicados** e **quantificadores**. A locução adjetiva "de primeira ordem" significa que suas linguagens podem se expressar sobre todos os **objetos** em um determinado domínio. Veja a figura abaixo.



Godel's World (retirado de Tarski's World)

Ela envolve figuras geométricas — *objetos* — de diferentes **formatos** e **tamanhos**. Suponha que queremos nos expressar que "Há um quadrado médio acima de todos os pentágonos" e "Há um quadrado e um triângulo grandes na mesma linha". Na lógica proposicional, ambas as sentenças seriam representadas por duas variáveis, digamos, x e y. Contudo, note três observações em comum: as duas sentenças discursam sobre os formatos e os tamanhos (**propriedades**) dos objetos — *predicados* —, as posições relativas entre objetos — *relações* — e uma quantificação ("há um", "todos"). Assim, podemos separá-las para nos tornarmos mais expressivos.

Por envolver predicados, a lógica de primeira ordem também é chamada Lógica de Predicados.

5.1.1 Alfabeto

A linguagem simbólica da lógica de predicados consiste em três tipos de símbolos:

Símbolos de objetos $x, a, h...$

Símbolos para predicados e relações $A(x), R(y,z)...$

Símbolos para funções de referência indireta $f(x), g(y,z)...$

Podemos usá-los para representar os conceitos a seguir.

Constantes e variáveis: funções de aridade zero

Constantes são símbolos usados para referenciar um, e apenas um, determinado objeto. Por exemplo, podemos referenciar o vértice raiz de uma árvore enraizada com uma constante, bem como o 0 no conjunto dos naturais. Variáveis, por outro lado, representam qualquer objeto.

Predicados e relações: funções proposicionais

Podemos usar esses símbolos para denotar alguma propriedade de objetos ou alguma relação entre objetos. De modo a introduzir uma notação simbólico-matemática para predicados e relações, o matemático Gottlob Frege mostrou como esses conceitos podem ser representados como funções proposicionais, cujas imagens resultam verdadeiro ou falso. Por exemplo:

Par: $\mathbb{N} \mapsto \{0,1\}$	Primos-entre-si: $\mathbb{N} \times \mathbb{N} \mapsto \{0,1\}$
$\text{Par}(15) = 0$	$\text{Primos-entre-si}(2,3) = 1$
$\text{Par}(3) = 0$	$\text{Primos-entre-si}(4,2) = 0$
$\text{Par}(4) = 1$	$\text{Primos-entre-si}(9,5) = 1$

Funções de referência indireta

Frege também definiu os símbolos para funções de referência indireta, que servem para referenciar um objeto a partir de outros. Por exemplo:

$$f : NOMES \mapsto NOMES$$
$$f(x) \equiv \text{feminino de } x$$
$$m(y) \equiv \text{masculino de } y$$

Claro, ainda possuímos os mesmos operadores lógicos. Além disso, temos mais dois símbolos, que representam os **quantificadores**.

Universal ($\forall$) - todo objeto satisfaz uma condição;

Existencial ($\exists$) - pelo menos um objeto satisfaz uma condição.

5.2 Estrutura Matemática

Ao contrário da lógica proposicional, onde há um tipo de símbolo para cada sentença — cada sentença é uma variável diferente —, na lógica de primeira ordem temos três tipos de símbolos para representar sentenças, o que a torna incompatível com a noção de valoração-verdade. Para prosseguirmos, precisamos tomar o conceito de **estrutura matemática**.

Definição 5.2.1 — Estrutura Matemática. Uma estrutura matemática é composta por quatro componentes:

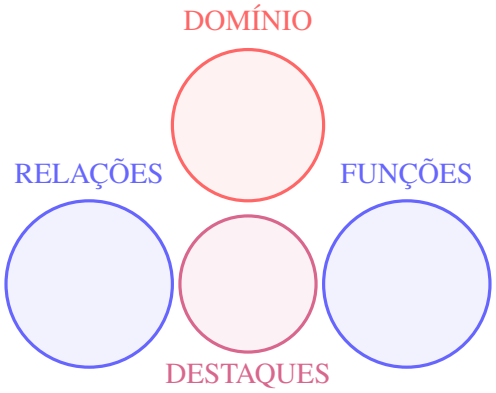
Domínio ou Universo Coleção de objetos que habitam a estrutura.

Destaques Subcoleção do domínio que contém as constantes da estrutura.

Relações Coleção de relações sobre o domínio, cada uma com sua aridade.

Funções Coleção de funções sobre o domínio, cada uma com sua aridade.

Podemos representar estruturas usando Diagramas de Venn:



5.2.1 Assinatura

Como vimos, usamos símbolos pra representar constantes, relações e funções. Devemos então, definir a linguagem simbólica para uma estrutura. Para isso, temos a assinatura e a linguagem. Enquanto a assinatura determina a quantidade de símbolos, a linguagem determina propriamente os mesmos. Podemos unir os dois conceitos, a fim de simplificação.

Definição 5.2.2 — Assinatura. Seja A uma estrutura matemática. A **assinatura** de A é definida pelos seguintes componentes:

1. Quantidade de destaques;
2. Quantidade de relações n-árias, onde $n \in \mathbb{N}$;
3. Quantidade de funções n-árias, onde $n \in \mathbb{N}$;

Dizemos que uma estrutura com assinatura L é uma **L-Estrutura**. Observe que uma assinatura pode ser compartilhada por várias estruturas simultaneamente, sem necessariamente estas serem iguais.

Definição 5.2.3 — Linguagem. Seja A uma estrutura matemática. A **linguagem** de A toma por base sua assinatura e é constituída de:

1. Um símbolo de constante para cada destaque de A;
2. Um símbolo de relação n-ária para cada relação n-ária de A, onde $n \in \mathbb{N}$;
3. Um símbolo de função n-ária para cada função n-ária de A, onde $n \in \mathbb{N}$;

5.2.2 Interpretação

Finalmente, uma vez definida uma linguagem simbólica, devemos fazer uma interpretação — a que o símbolo corresponde em uma estrutura. Assim:

Definição 5.2.4 — Interpretação. Seja A uma L-Estrutura. A interpretação da assinatura L na estrutura A é uma função que associa cada símbolo de L a um elemento de cada componente de A. Portanto,

1. A cada símbolo de constante c de L, é associado um destaque de A — c^A;
2. A cada símbolo de relação n-ária R de L, é associado uma relação n-ária de A — R^A;
3. A cada símbolo de função n-ária f de L, é associado uma função n-ária de A — f^A;

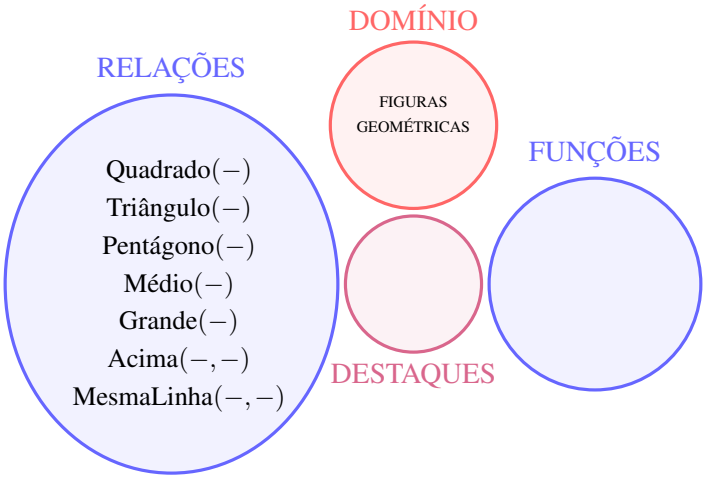
■ **Exemplo 5.1** Defina uma estrutura A, a assinatura L de A e uma interpretação de L em A para ser possível representar matematicamente as duas sentenças abaixo (do exemplo da seção 5.1).

Vamos fazer uma análise:

1. **Há um quadrado médio acima de todos os pentágonos.**
   - *quadrado*, *pentágono* e *médio* são predicados.
   - *acima* é uma relação entre objetos.

2. **Há um quadrado e um triângulo grandes na mesma linha.**
   - *quadrado*, *triângulo* e *grande* são predicados.
   - *na mesma linha* é uma relação entre objetos.

Assim, construímos nossa estrutura baseado no que coletamos:



Logo, definimos a assinatura L de A desta maneira:

- 0 constantes;
- 5 símbolos de relação unária: $Q(-), T(-), P(-), M(-), G(-)$;
- 2 símbolos de relação binária: $A(-,-), L(-,-)$;
- 0 funções;

Finalmente, devemos definir a que corresponde cada símbolo: uma interpretação.

- $Q^A(-)$: Quadrado$(-)$;
- $T^A(-)$: Triângulo$(-)$;
- $P^A(-)$: Pentágono$(-)$;
- $M^A(-)$: Médio$(-)$;
- $G^A(-)$: Grande$(-)$;
- $A^A(-,-)$: Acima$(-,-)$;
- $L^A(-,-)$: MesmaLinha$(-,-)$;

Estas seriam as duas sentenças representadas na lógica de primeira ordem:

1. $\exists x(Q^A(x) \wedge M^A(x) \wedge \forall y(P^A(y) \rightarrow A^A(x,y)))$
2. $\exists x \exists y(Q^A(x) \wedge G^A(x) \wedge T^A(y) \wedge G^A(y) \wedge L^A(x,y))$

■

5.3 Funções entre estruturas

Podemos fazer uma "comunicação" entre estruturas através de **funções de mapeamento**, isto é, cada elemento de uma estrutura tem um correspondente em outra mediante a função (chamamos isso de **preservação de componentes**). O tipo de função de mapeamento mais básico é o **homomorfismo**.

Definição 5.3.1 — Homomorfismo. Sejam A e B estruturas de uma mesma assinatura L. Uma função $h : dom(A) \mapsto dom(B)$ é dita **homomorfismo** se, e somente se:

1. Para todo símbolo de constante c de L,
   $h(c^A) = c^B$ *(Preserva destaques)*;
2. Para todo símbolo de relação n-ária R de L e toda n-upla $(a_1,...,a_n) \in dom(A)$,
   $(a_1,...,a_n) \in R^A \rightarrow (h(a_1),...,h(a_n)) \in R^B$ *(Preserva relações:* $R^A \subseteq R^B \cap A^n$*)*;
3. Para todo símbolo de função n-ária f de L e toda n-upla $(a_1,...,a_n) \in dom(A)$,
   $h(f^A(a_1,...a_n)) = f^B(h(a_1),...h(a_n))$ *(Preserva funções)*;

5.3.1 Imersão e variações

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

1. h é injetora;
2. Para todo símbolo de relação n-ária R de L e toda n-upla $(a_1,...,a_n) \in dom(A)$,
   $(a_1,...,a_n) \in R^A \leftrightarrow (h(a_1),...,h(a_n)) \in R^B$ *(Versão mais forte:* $R^A = R^B \cap A^n$*)*;

Isomorfismo

Uma imersão $h : dom(A) \mapsto dom(B)$ é dito **isomorfismo** se for sobrejetora.

Endomorfismo

Um homomorfismo $h : dom(A) \mapsto dom(A)$ é dito **endomorfismo**.

Automorfismo

Um isomorfismo $h : dom(A) \mapsto dom(A)$ é dito **automorfismo**.

5.4 Subestruturas

Dispondo de duas estruturas similares, queremos saber se uma é subestrutura da outra — subgrafos, por exemplo. Para tal, deve-se atender a duas condições.

Definição 5.4.1 — Subestrutura. Sejam A e B duas estruturas de mesma assinatura. Dizemos que **A é subestrutura de B** — notação $A \subseteq B$ — se, e somente se:

1. O domínio de A está contido no domínio de B ($dom(A) \subseteq dom(B)$);
2. A função identidade $i : dom(A) \mapsto dom(B) / i(x) = x$ é uma imersão.

■ **Exemplo 5.2** Mostre que $A \subseteq B$, definidas abaixo.

	A	**B**
Domínio	$\{1,2,10\}$	$\{1,2,5,10,13\}$
Destaques	$\{1\}$	$\{1\}$
Relações	Divide$(-,-)$	Menor-ou-igual$(-,-)$
Funções	MDC$(-,-)$	MDC$(-,-)$

Tomemos:
$c^A = 1$, $c^B = 1$;
$R^A(-,-)$: Divide$(-,-)$, $R^B(-,-)$: Menor-ou-igual$(-,-)$;
$f^A(-,-)$: MDC$(-,-)$, $f^B(-,-)$: MDC$(-,-)$.

Temos que $\{1,2,10\} \subseteq \{1,2,5,10,13\}$, assim, a 1ª condição é satisfeita;

Analisemos a função identidade. Ela, pela sua definição, é injetora.
Temos que $i(c^A) = c^B$ (1 = 1); Assim, i **preserva destaques**.
Vejamos as duplas de A: $A^2 = \{(1,1),(1,2),(1,10),(2,1),(2,2),(2,10),(10,1),(10,2),(10,10))\}$.
Temos que:
$R^A = \{(1,1),(1,2),(1,10),(2,2),(2,10),(10,10)\}$
$R^B = \{(1,1),(1,2),(1,5),(1,10),(1,13),(2,2),(2,5),(2,10),(2,13),(10,10),(10,13),(13,13)\}$
Como $R^A = R^B \cap A^2$, i **preserva relações**.
Também temos que:

$$i(f^A(1,1)) = f^B(i(1),i(1)) = 1 \quad i(f^A(1,2)) = f^B(i(1),i(2)) = 1 \quad i(f^A(1,10)) = f^B(i(1),i(10)) = 1$$
$$i(f^A(2,1)) = f^B(i(2),i(1)) = 1 \quad i(f^A(2,2)) = f^B(i(2),i(2)) = 2 \quad i(f^A(2,10)) = f^B(i(2),i(10)) = 2$$
$$i(f^A(10,1)) = f^B(i(10),i(1)) = 1 \quad i(f^A(10,2)) = f^B(i(10),i(2)) = 2 \quad i(f^A(10,10)) = f^B(i(10),i(10)) = 10$$

Assim, i **preserva funções**. Logo, i é uma imersão e a 2ª condição é satisfeita.

Desse modo, $A \subseteq B$. ■

5.4.1 A menor subestrutura

Dada uma L-Estrutura A e um conjunto X de elementos do domínio de A, tomemos a menor subestrutura que contém X em seu domínio. Assim, desejamos construir, **de forma mínima**, uma estrutura B tal que $B \subseteq A$ e $X \subseteq dom(B)$. A notação é $< X >_A$. Este é um problema de otimização.

Precisamos adicionar à estrutura B as mesmas funções, constantes e relações de A, e adicionar então ao domínio de B, que inicia com X, os elementos de A necessários para que $B \subseteq A$.

Se X é finito e $< X >_A = A$ (não há subestrutura de A menor que a própria A), ela é dita **finitamente gerada**.

■ **Exemplo 5.3** Dado $X = \{2,6\}$ e a estrutura A abaixo, determine $B =< X >_A$.

Domínio $\{0,1,2,3,4,5,6,9,11,15,20\}$

Destaques $\{0,5,11\}$

Relações Divide$(-,-)$

Funções Quadrado-mod-10$(-)$

Tomemos f^A: Quadrado-mod-10$(-)$. As mesmas constantes, relações e funções são adicionadas a B. Assim, veremos a progressão do domínio:

Iniciamos com: $\{2,6\}$
Para manter consisa a definição, precisamos adicionar os destaques ao domínio.
Adição dos destaques: $\{0,2,5,6,11\}$
Finalmente, precisamos fazer com que B seja subestrutura de A. Note, por exemplo, que $f^B(2) = 4$, que não está no domínio. Assim, para fazermos $i(f^B(2)) = f^A(i(2))$, precisamos adicioná-lo. Repetir essa análise em todos os elementos causa:
Preservando componentes: $\{0,1,2,4,5,6,11\}$

Assim, $< X >_A$ é a estrutura com mesmos componentes que A e de domínio $\{0,1,2,4,5,6,11\}$. ■

5.5 Extensão

Quando $< \emptyset >_A = A$, todos os elementos do domínio de A são acessíveis a partir dos destaques e funções de A. Caso contrário, podemos estender o conjunto de destaques com novos quantos forem os elementos inacessíveis. Essa nova estrutura B é dita **extensão** de A e A é dita **reduto** de B.

6. Sintaxe

De modo similar à Lógica Proposicional, estudaremos o processo de **formação** das expressões da lógica de predicados. No capítulo seguinte, veremos a semântica.

6.1 Alfabeto

Nós já abordamos o alfabeto da lógica de primeira ordem. Mas, uma vez que vimos estruturas, podemos introduzir mais um conceito: símbolos que não mudam seu significado, como os operadores, quantificadores e variáveis, são chamados **símbolos lógicos**. Por outro lado, símbolos de constantes, relações e funções, os quais fazem parte do alfabeto, mas precisamos criar uma assinatura para introduzí-los e uma interpretação para terem sentido, são chamados **símbolos não lógicos**.

6.2 Termos e Fórmulas Atômicas

A formação de expressões se dá a partir da reunião de termos e fórmulas atômicas. Informalmente, **termos** são representações de objetos, enquanto **fórmulas atômicas** são representações de relações.

Podemos definir o conjunto de todos os termos de uma assinatura indutivamente.

Definição 6.2.1 — Conjunto dos Termos. Seja L uma linguagem. O **conjunto dos termos** $TERMOS_L$ de L é o menor conjunto de expressões sobre o vocabulário simbólico de L tal que:

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

Se um termo não contém variáveis, ele é dito **fechado**.

Definição 6.2.2 — Fórmula Atômica. Seja L uma linguagem. Uma **fórmula atômica** de L 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.

Se uma fórmula atômica não contém variáveis, ela é dita **sentença atômica**.

6.2.1 Fórmulas bem formadas

De modo similar à lógica proposicional, existem palavras sobre o alfabeto simbólico de uma assinatura que não são **legítimas**. Podemos definir o conjunto de expressões legítimas — fórmulas bem formadas ($FORM$) — indutivamente:

- toda fórmula atômica é uma fórmula bem formada;
- se ω é uma fórmula bem formada, então $\neg\omega$ também é;
- se ω_1 e ω_2 são fórmulas bem formadas, então $(\omega_1 \wedge \omega_2)$ também é;
- se ω_1 e ω_2 são fórmulas bem formadas, então $(\omega_1 \vee \omega_2)$ também é;
- se ω_1 e ω_2 são fórmulas bem formadas, então $(\omega_1 \rightarrow \omega_2)$ também é;
- se ω é uma fórmula bem formada, então $\exists x\omega(x)$ também é;
- se ω é uma fórmula bem formada, então $\forall x\omega(x)$ também é;

$\omega(x)$ *significa que a variável x ocorre em* ω.

6.3 Variáveis

Podemos fazer um estudo preciso das variáveis que ocorrem em uma fórmula. Lembre: variáveis são termos que representam qualquer objeto.

6.3.1 Variáveis livres e ligadas

Variável livre é uma variável fora da ação de um quantificador. Por exemplo, na fórmula abaixo,

$$\forall x(R(x,y))$$

y é uma variável livre, enquanto x é uma **variável ligada**. Podemos definir o conjunto de variáveis livres em uma fórmula usando uma função recursiva:

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

■ **Exemplo 6.1** Determine o conjunto de variáveis livres de $\exists x(R(x,y) \rightarrow \forall y(P(z,y) \wedge \exists w(S(w,u) \vee \neg R(w,y))))$.

Tomemos $VL(\exists x(R(x,y) \rightarrow \forall y(P(z,y) \wedge \exists w(S(w,u) \vee \neg R(w,y)))))$.
$= VL((R(x,y) \rightarrow \forall y(P(z,y) \wedge \exists w(S(w,u) \vee \neg R(w,y))))) - \{x\}$
$= (VL(R(x,y)) \cup VL(\forall y(P(z,y) \wedge \exists w(S(w,u) \vee \neg R(w,y))))) - \{x\}$

$= (\{x,y\} \cup (VL(\forall y(P(z,y))) \cup VL(\exists w(S(w,u) \vee \neg R(w,y))))) - \{x\}$
$= (\{x,y\} \cup ((VL(P(z,y)) - \{y\}) \cup (VL(S(w,u) \vee \neg R(w,y)) - \{w\}))) - \{x\}$
$= (\{x,y\} \cup ((\{z,y\} - \{y\}) \cup ((VL(S(w,u)) \cup VL(\neg R(w,y))) - \{w\}))) - \{x\}$
$= (\{x,y\} \cup (\{z\} \cup ((\{w,u\} \cup VL(R(w,y))) - \{w\}))) - \{x\}$
$= (\{x,y\} \cup (\{z\} \cup ((\{w,u\} \cup \{w,y\}) - \{w\}))) - \{x\}$
$= (\{x,y\} \cup (\{z\} \cup (\{w,u,y\} - \{w\}))) - \{x\}$
$= (\{x,y\} \cup (\{z\} \cup \{u,y\})) - \{x\}$
$= (\{x,y\} \cup \{z,u,y\}) - \{x\}$
$= \{x,y,z,u\} - \{x\}$
$= \{y,z,u\}.$ ∎

6.3.2 Substituição de variáveis

Em uma interpretação, definimos as constantes e as funções, então a semântica dos termos fechados é definida imediatamente. Para termos abertos, se quisermos definir a que objeto uma variável se refere, usamos uma substituição. Valendo-se do fato de que o conjunto dos termos de uma linguagem é livremente gerado, podemos definir recursivamente uma função que realiza a substituição de variáveis por termos (vamos dividí-la em três partes).

Em termos

Definimos a função de substituição $[s/x]$ — "s entra no lugar de x" — remove todas as ocorrências de x em um termo e põe s em seu lugar recursivamente:
$[./.] : TERMOS_L \times TERMOS_L \times$ VARIÁVEIS $\mapsto TERMOS_L$
$x[s/y] = s$, se $x = y$
$x[s/y] = x$, se $x \neq y$
$c[s/y] = c$
$f(t_1,...,t_n)[s/y] = f(t_1[s/y],...,t_n[s/y])$

Em fórmulas atômicas

$[./.] : FORM \times TERMOS_L \times$ VARIÁVEIS $\mapsto FORM$
$R(t_1,...,t_n)[s/y] = R(t_1[s/y],...,t_n[s/y])$
$(t_1 = t_2)[s/y] = (t_1[s/y] = t_2[s/y])$

Em fórmulas não atômicas

$[./.] : FORM \times TERMOS_L \times$ VARIÁVEIS $\mapsto FORM$
$(\neg\psi)[s/y] = \neg(\psi[s/y])$
$(\rho \square \theta)[s/y] = (\rho[s/y] \square \theta[s/y])$, onde $\square \in \{\vee, \wedge, \rightarrow\}$
$(\square x\omega)[s/y] = \square x\omega[s/y]$, onde $\square \in \{\exists, \forall\}$

É muito importante ressaltar que, ao fazermos uma substituição, a natureza da variável **deve persistir**. Ou seja, se ela é ligada, deve permanecer ligada, e, se for livre, deve permanecer livre. Veja o exemplo abaixo.

■ **Exemplo 6.2** Faça a substituição $[y/x]$ em $\exists y(R(x,y) \wedge \forall x(P(z,x)))$.

Tomemos $(\exists y(R(x,y) \wedge \forall x(P(z,x))))[y/x]$.
$= \exists y(R(x,y) \wedge \forall x(P(z,x)))[y/x]$
$= \exists y(R(x,y)[y/x] \wedge \forall x(P(z,x))[y/x])$
$= \exists y(R(x[y/x],y[y/x]) \wedge \forall x(P(z,x)[y/x]))$
$= \exists y(R(y,y) \wedge \forall x(P(z[y/x],x[y/x])))$
$= \exists y(R(y,y) \wedge \forall x(P(z,y)))$. ∎

Note que $\forall x(P(z,y))$ não é uma fórmula bem formada e $R(y,y)$ está com um significado diferente, graças ao $\exists y$. Assim, podemos definir quando uma substituição pode ser feita com a definição de **termo livre**:

Definição 6.3.1 — Termo livre. 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 x em ψ;
- ϕ é da forma $(\rho\square\theta)$ e t está livre para x em ρ e em θ, onde $\square \in \{\vee,\wedge,\rightarrow\}$;
- ϕ é da forma $(\square y\omega)$ e $x = y$ ou y não ocorre em t e t está livre para x em ω, onde $\square \in \{\exists,\forall\}$.

No nosso exemplo, $x \neq y$ e y ocorre em t. Assim, a substituição já seria interrompida na primeira recursão.

7. Semântica

Veremos como podemos valorar expressões na lógica de predicados (lembre que valoração-verdade não é possível de ser aplicada) e algumas estruturas especiais relacionadas à verdade das expressões.

7.1 Termos e Fórmulas Atômicas

Para termos fechados, seu significado é definido diretamente pela **interpretação** de uma assinatura L em uma L-Estrutura A:

- Se t for uma constante, então $t^A = c^A$;
- Se t for um termo composto $f(t_1,...,t_n)$, então $t^A = f^A(t_1^A,...t_n^A)$;

Já para termos abertos, ao quisermos definir a que objeto as variáveis referenciam, precisamos fazer uma substituição de variáveis: $t^A = t[s_1/x_1,...,s_n/x_n]$.

Sabendo o significado dos termos, podemos definir quando fórmulas atômicas são verdadeiras. Para sentenças atômicas:

- $R^A(t_1^A,...,t_n^A)$ é verdadeira $\leftrightarrow (t_1^A,...,t_n^A) \in R^A$
- $(t_1^A = t_2^A)$ é verdadeira $\leftrightarrow t_1$ e t_2 forem o mesmo elemento do domínio de A.

Fazendo a mesma análise para fórmulas atômicas abertas:

- $R^A(t_1^A,...,t_n^A)$ é verdadeira $\leftrightarrow (t_1^A[s_1/x_1,...,s_n/x_n],...,t_n^A[s_1/x_1,...,s_n/x_n]) \in R^A$
- $(t_1^A = t_2^A)$ é verdadeira $\leftrightarrow t_1^A[s_1/x_1,...,s_n/x_n]$ e $t_2^A[s_1/x_1,...,s_n/x_n]$ forem o mesmo elemento do domínio de A.

7.1.1 Modelo e contramodelo

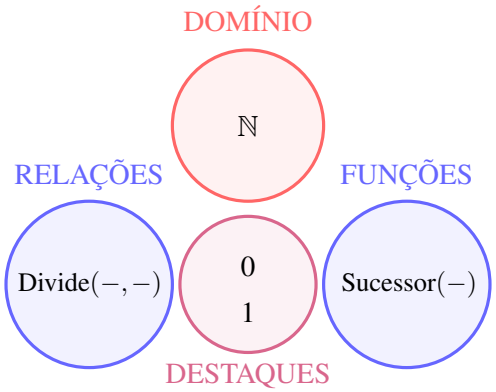
Ao valorar uma sentença atômica em uma estrutura, podemos classificar esta como ou **modelo**, ou **contramodelo**, ou ambos.

Definição 7.1.1 — Modelo. Seja L uma assinatura, A uma L-Estrutura e α uma sentença atômica de L.

- Se existe pelo menos uma interpretação tal que α^A seja verdadeira, A é dita **modelo para** α.
- Se existe pelo menos uma interpretação tal que α^A seja falsa, A é dita **contramodelo para** α.

7.2 Modelo Canônico

Veja a estrutura A abaixo.



Tomemos: $f^A(-)$: Sucessor$(-)$, $R^A(-,-)$: Divide$(-,-)$, $b^A = 0$, $c^A = 1$.

Note que $R^A(c^A, b^A)$ é verdadeira, bem como $R^A(c^A, f^A(c^A))$ e $R^A(c^A, f^A(f^A(c^A)))$. Conseguimos, nesse exemplo, graças à infinitude do domínio, criar infinitas sentenças atômicas que sejam verdadeiras em A, e agrupá-las em um conjunto especial.

Definição 7.2.1 — Diagrama Positivo. Seja L uma assinatura e A uma L-Estrutura. O conjunto de todas as sentenças atômicas de L que são verdadeiras em A — ou seja, as quais A é modelo —, é dito **diagrama positivo de A**, e a notação é $diag^+(A)$.

Caso $< \emptyset >_A \neq A$, **usamos a extensão de A para gerar o diagrama positivo**.

Para esse exemplo, $diag^+(A) = \{b = b, c = c, f(b) = f(b), f(b) = c, ..., R(c,b), R(c, f(c)), R(c, f(f(c))), R(f(c), f(f(f(c)))), ...\}$.

O que fizemos chama-se processo **de estruturas para sentenças atômicas**. Agora, desejamos fazer o inverso: **de sentenças atômicas para estruturas**.

Suponha um conjunto de sentenças atômicas quaisquer. Queremos construir uma estrutura, a **mais genérica possível**, na assinatura dessas sentenças que seja modelo para todas essas. Essa estrutura chama-se **modelo canônico**.

Lema 7.2.1 Todo conjunto de sentenças atômicas possui um modelo canônico.

7.2.1 A relação $\sim$

Para definirmos o modelo canônico de um conjunto de sentenças atômicas, precisamos definir seu domínio, conjunto de relações, de destaques e de funções.

Inicialmente, o domínio é o conjunto de todos os termos fechados que ocorrem no conjunto de sentenças (para o exemplo acima, teríamos $\{b, c, f(b), f(c), f(f(c))...\}$). Porém, note que c e $f(b)$ são iguais. Isso nos leva a um problema: o domínio do modelo canônico pode ser maior que o da estrutura original, o que é impossível.

Desse modo, precisamos **agrupar os termos semelhantes em classes de equivalência**. Para tal, definamos uma relação $\sim$:

Definição 7.2.2 — Relação $\sim$. Seja T o diagrama positivo de alguma estrutura.
$\sim = \{(s,t) \in \sim$ se, e somente se, $(s = t) \in T\}$

Agora demonstremos que $\sim$ é uma relação de equivalência. Se pudermos fazer isso, podemos usar apenas o termo fechado que é **representante** da classe gerada, ao invés de todos os termos fechados.

Demonstração. Queremos provar que $\sim$ é reflexiva, simétrica e transitiva.

Reflexividade Como, para todo termo t, $(t = t) \in T$, $(t,t) \in \sim$. Assim, $\sim$ é reflexiva.

Simetria Suponha dois termos s e t tal que $s = t$. Então, $(s = t) \in T$ e, pela comutatividade da igualdade, $(t = s) \in T$. Logo, se $(s,t) \in \sim$, então $(t,s) \in \sim$. Assim, $\sim$ é simétrica.

Transitividade Suponha três termos s, t e r tal que $s = t$ e $t = r$. Então, $(s = t) \in T$, $(t = r) \in T$ e, pela transitividade da igualdade, $(s = r) \in T$. Assim, se $(s,t) \in \sim$ e $(t,r) \in \sim$, então $(s,r) \in \sim$. Logo, $\sim$ é transitiva.

$\sim$ é uma relação de equivalência. ■

7.2.2 Obtenção do modelo canônico

Mostraremos o passo-a-passo para a obtenção de um modelo canônico A de um conjunto de sentenças atômicas. Seja L uma assinatura.

Domínio

Como dissemos anteriormente, ao invés de definirmos como o conjunto de todos os termos fechados, usaremos apenas o representante da classe do termo fechado — o denotaremos por $t^{\sim}$. No exemplo anterior, teríamos:

$$\{b^{\sim}, c^{\sim}, f(c)^{\sim}, f(f(c))^{\sim}, ...\}$$

Formalmente: O domínio do modelo canônico é o conjunto das classes de equivalência dos elementos de um conjunto de termos fechados X sobre a relação $\sim$.

Destaques

O conjunto de destaques é subconjunto do domínio. Assim, de forma similar:

$$\{b^{\sim}, c^{\sim}\}$$

Formalmente: Para cada símbolo de constante c de L, $c^{\sim}$ será uma constante do modelo canônico.

Relações

O modelo canônico tem as mesmas relações que ocorrem no conjunto de sentenças atômicas. Porém, a fim de generalizar a estrutura, usaremos apenas os símbolos.

$$\{R(-,-)\}$$

Formalmente: Para cada símbolo de relação n-ária de L, $(t_1^\sim,...,t_n^\sim) \in R^A$ se, e somente se, $R(t_1,...,t_n)$ pertence ao conjunto de sentenças atômicas.

Funções

Mesma maneira que o conjunto de relações, mas tendo em mente que aplicações de funções em termos também são termos.

$$\{f^\sim(-)\}$$

Formalmente: Para cada símbolo de função n-ária de L, $f^A(t_1^\sim,...,t_n^\sim) = f(t_1,...,t_n)^\sim$.

Desejamos uma estrutura a mais genérica possível para que, ao encontrar qualquer estrutura que também satisfaça a descrição da primeira, seja possível definir um homomorfismo entre as duas.

■ **Exemplo 7.1** Dada a estrutura A abaixo, determine seu diagrama positivo.

Domínio $\{0,1,2,3\}$

Destaques $\{3\}$

Relações Maior-que$(-,-)$, Ímpar$(-)$

Funções Quadrado-mod-4$(-)$

Note que Quadrado-mod-4 não nos permite acessar os elementos 0 e 2 do domínio usando o destaque disponível. Assim, $< \emptyset >_A \neq A$ e, portanto, usaremos a extensão de A:
Tomando a assinatura:

- 3 símbolos de destaque: k, b, c;
- 1 símbolo de relação unária: $I(-)$;
- 1 símbolo de relação binária: $M(-,-)$;
- 1 símbolo de função unária: $f(-)$;

E a interpretação:
$k^A = 3, b^A = 0, c^A = 2$;
$I^A(-)$: Ímpar$(-)$, $M^A(-,-)$: Maior-que$(-,-)$;
$f^A(-)$: Quadrado-mod-4$(-)$.

Temos que, portanto:

$$diag^+(A) = \{I(k), I(f(k)), M(k,c), M(k,f(k)), M(k,b), M(c,f(k)), M(c,b), M(f(k),b), k = k, f(k) = f(k), b = b, c = c, f(k) = f(f(k))\}$$

Perceba que podemos aumentar nosso diagrama positivo acrescentando a ele, por exemplo, $f(k) = f(f(f(k)))$*, mas seria redundância.* ■

■ **Exemplo 7.2** Determine o modelo canônico do diagrama positivo do exemplo anterior.

Apesar de A ser modelo para todas as sentenças, A não é a mais genérica possível. Assim:

Domínio $\{k^{\sim}, b^{\sim}, c^{\sim}, f(k)^{\sim}\}$

Destaques $\{k^{\sim}, b^{\sim}, c^{\sim}\}$

Relações $\{I(-), M(-,-)\}$

Funções $\{f^{\sim}(-)\}$

Observe que $f(f(k))$ pertence à mesma classe de equivalência que $f(k)$. ■

7.3 Modelo de semântica de Tarski

Baseado no modelo de verdade definido pelo matemático Alfred Tarski, definimos recursivamente uma função sobre o conjunto FORM que retorna o valor-verdade de uma fórmula sobre uma estrutura, sujeita a uma interpretação em sua assinatura e a possivelmente sob uma dada função de substituição de variáveis livres.

Seja L uma assinatura, A uma L-Estrutura e ϕ uma sentença de L. O valor-verdade de ϕ é definido da seguinte forma:

1. ϕ é atômica:
   - ϕ da forma $R(t_1,...,t_n)$: ϕ^A é verdadeira se, e somente se, $(t_1^A,...,t_n^A) \in R$;
   - ϕ da forma $(t_1 = t_2)$: ϕ^A é verdadeira se, e somente se, t_1^A e t_2^A forem o mesmo elemento do domínio de A.
2. ϕ é da forma $(\neg\psi)$:
   $(\neg\psi)^A$ é verdadeira se, e somente se, ψ^A não for verdadeira.
3. ϕ é da forma $(\rho \wedge \theta)$:
   $(\rho \wedge \theta)^A$ é verdadeira se, e somente se, ρ^A é verdadeira e θ^A é verdadeira.
4. ϕ é da forma $(\rho \vee \theta)$:
   $(\rho \vee \theta)^A$ é verdadeira se, e somente se, ρ^A é verdadeira ou θ^A é verdadeira.
5. ϕ é da forma $(\rho \rightarrow \theta)$:
   $(\rho \rightarrow \theta)^A$ é verdadeira se, e somente se, se ρ^A é verdadeira, então θ^A também é.
6. ϕ é da forma $(\exists\omega)$:
   $(\exists\omega)^A$ é verdadeira se, e somente se, existe um $a \in A$ tal que $\omega^A[a/x]$ é verdadeira.
7. ϕ é da forma $(\forall\omega)$:
   $(\forall\omega)^A$ é verdadeira se, e somente se, para todo $a \in A$, $\omega^A[a/x]$ é verdadeira.

8. O Problema da Satisfatibilidade

Dada uma fórmula ϕ, pergunta-se: ***ϕ é satisfatível?***

Voltamos ao nosso cobiçado problema computacional. Porém, ao invés de mostrarmos cinco métodos para resolvê-los, mostraremos o mais "popular" — o método da Resolução — e mais um que serviu de base para todos os seguintes. Antes disso, devemos fazer mais uma análise de sintaxe sobre as fórmulas, preliminar para se poder resolver uma instância do problema, bem como de um subproblema deste envolvendo termos: o problema da unificação.

8.1 Satisfatibilidade

φ é satisfatível se existe uma L-Estrutura A e uma interpretação de L em A tal que A satisfaz φ.

φ é refutável se existe uma L-Estrutura A e uma interpretação de L em A tal que A refuta φ.

φ é tautologia ou válida se para toda L-Estrutura A e toda interpretação de L em A, A satisfaz φ.

φ é insatisfatível se para toda L-Estrutura A e toda interpretação de L em A, A refuta φ.

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

Um conjunto de sentenças Γ é satisfatível se existe uma L-Estrutura A e uma interpretação de L em A tal que A satisfaz todas as sentenças de Γ.

φ é consequência lógica de Γ se para toda L-Estrutura A e toda interpretação de L em A, se A satisfaz Γ, então A também satisfaz φ (notação: $\Gamma \models \varphi$).

8.2 Sintaxe das entradas

8.2.1 Forma Normal Prenex

Quando uma fórmula da lógica de primeira ordem tem uma divisão entre seus quantificadores e o restante da fórmula, dizemos que ela se encontra na **forma normal prenex**:

$$\underbrace{Q_1x_1...Q_nx_n}_{\textbf{Prefixo}} \quad \underbrace{(M)}_{\textbf{Matriz}}$$

O prefixo contém todos os quantificadores da fórmula, enquanto a matriz é o restante — não possui quantificadores. Para que seja possível transformar uma fórmula para a FNP, introduzimos mais 7 equivalências lógicas *(Q representa um quantificador)*:

$$\begin{array}{lrcl}
(i) & Qx(\varphi(x)) \wedge \psi & \equiv & Qx(\varphi(x) \wedge \psi) \\
(ii) & Qx(\varphi(x)) \vee \psi & \equiv & Qx(\varphi(x) \vee \psi) \\
(iii) & Qx(\varphi(x)) & \equiv & Qy(\varphi(y)) \\
(iv) & \neg\forall x(\varphi) & \equiv & \exists x(\neg\varphi) \\
(v) & \neg\exists x(\varphi) & \equiv & \forall x(\neg\varphi) \\
(vi) & \forall x(\varphi) \wedge \forall y(\psi) & \equiv & \forall x\forall y(\varphi \wedge \psi) \\
(vii) & \exists x(\varphi) \vee \exists y(\psi) & \equiv & \exists x\exists y(\varphi \vee \psi)
\end{array}$$

A *(iii)* nos diz que podemos mudar a variável do quantificador e suas variáveis ligadas. Isso é útil quando a fórmula possui uma ocorrência da mesma variável e não podemos aplicar uma das duas últimas equivalências lógicas.

■ **Exemplo 8.1** Transforme $\phi = (\forall xP(x) \to \exists xQ(x))$ para a FNP.

Temos que $\phi \equiv (\neg\forall xP(x) \vee \exists xQ(x))$
$\equiv (\exists x\neg P(x) \vee \exists xQ(x))$
$\equiv \exists x(\neg P(x) \vee Q(x))$. ■

■ **Exemplo 8.2** Transforme $\phi = (\forall x(P(x) \wedge \exists y\neg R(y,z)) \to \exists x\forall yQ(x,y,z))$ para a FNP.

Temos que $\phi \equiv (\neg\forall x(P(x) \wedge \exists y\neg R(y,z)) \vee \exists x\forall yQ(x,y,z))$
$\equiv (\exists x\neg(P(x) \wedge \exists y\neg R(y,z)) \vee \exists x\forall yQ(x,y,z))$
$\equiv (\exists x(\neg P(x) \vee \neg\exists y\neg R(y,z)) \vee \exists x\forall yQ(x,y,z))$
$\equiv (\exists x(\neg P(x) \vee \forall y\neg\neg R(y,z)) \vee \exists x\forall yQ(x,y,z))$
$\equiv (\exists x(\neg P(x) \vee \forall yR(y,z)) \vee \exists x\forall yQ(x,y,z))$
$\equiv \exists x(\neg P(x) \vee \forall yR(y,z) \vee \forall yQ(x,y,z))$
$\equiv \exists x(\neg P(x) \vee \forall yR(y,z) \vee \forall wQ(x,w,z))$ *(Mudança de variável.)*
$\equiv \exists x\forall y\forall w(\neg P(x) \vee R(y,z) \vee Q(x,w,z))$. ■

8.2.2 Forma Normal de Skolem

Uma fórmula na lógica de predicados sem quantificadores existenciais é dita estar na **forma normal de Skolem**, em homenagem ao matemático dinamarquês Thoralf Skolem, que definiu um método que permite retirá-los de qualquer fórmula: o método da **skolemização**. O teorema a seguir, definido por ele e pelo matemático alemão Leopold Löwenheim, afirma esse método.

Teorema 8.2.1 — Teorema de Löwenheim-Skolem. Seja φ uma fórmula na lógica de predicados numa assinatura L, tal que φ está na forma normal prenex. Seja φ' a fórmula resultante da eliminação dos quantificadores existenciais que ocorrem em φ, e cujas variáveis correspondentes são substituídas por termos do tipo $f(x_1,...,x_n)$, onde f é um 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 A' que é modelo para φ' simplesmente acrescentando à estrutura A uma interpretação para cada novo símbolo de função em A'.

Caso não hajam quantificadores universais anteriores ao existencial, as variáveis são substituídas por símbolos de constantes (funções de aridade zero) e na estrutura A é acrescentado destaques correspondentes.

■ **Exemplo 8.3** Transforme $\phi = \forall x \forall y \exists z(R(x,z) \rightarrow P(y))$ para a FNS.

ϕ já está na FNP, então podemos aplicar a skolemização.
Como há 2 quantificadores universais anteriores à $\exists z$, adicionamos uma função f que depende de x e y, as variáveis quantificadas, e eliminamos o existencial. f é dita **função de Skolem**. Assim, temos:

$$\forall x \forall y(R(x, f(x,y)) \rightarrow P(y))$$

■

■ **Exemplo 8.4** Transforme $\phi = \exists x \forall y \forall z \exists u \exists v \forall w(R(x,y,z,u,v,w) \vee \exists t(P(t)))$ para a FNS.

ϕ não está na FNP. Transformando-a:

$$\exists x \forall y \forall z \exists u \exists v \forall w \exists t(R(x,y,z,u,v,w) \vee (P(t)))$$

Agora podemos aplicar a skolemização. Para cada existencial, analisaremos os universais anteriores a ele e acrescentaremos uma função ou constante:

Existencial	Quant. universais anteriores
$\exists x$	0
$\exists u$	2
$\exists v$	2
$\exists t$	3

Assim, temos 1 **constante de Skolem** e 3 funções de Skolem. Adicionando c, $f(-,-)$, $g(-,-)$, e $h(-,-,-)$, temos:

$$\forall y \forall z \forall w(R(c,y,z,f(y,z),g(y,z),w) \vee P(h(y,z,w)))$$

Importante! As constantes e funções de Skolem não podem já pertencer à assinatura, e devem ser diferentes para cada existencial. ■

8.3 Unificação de Termos

Seja L uma assinatura e t_1 e t_2 termos de L nos quais ocorrem as variáveis $x_1, ...x_n$. O problema da unificação surge ao tentar encontrar uma substituição do tipo $[s_1/x_1, ..., s_n/x_n]$ tal que sua aplicação fará t_1 e t_2 serem idênticos. Caso haja, t_1 e t_2 são ditos **unificáveis**. Assim, o problema se apresenta como: *Dados dois termos t_1 e t_2, t_1 e t_2 são unificáveis?*

Em 1930, o matemático francês Jacques Herbrand definiu um método algorítmico simples para resolver o problema utilizando regras de transformação de sistema de equações. Herbrand mostrou que o método era correto e completo em relação ao conjunto de soluções.

Antes de definir as regras de transformação, vejamos alguns conceitos básicos:

- Uma **equação** é um par de termos $(s = t)$.
- Se s e t forem fechados, $s = t$ é dita **básica**.
- Uma substituição θ é chamada de **unificadora padrão** de uma equação se $\theta(s) = \theta(t)$.
- Um sistema de equações é um conjunto de equações e uma substituição θ é uma **unificadora** de S se ela unifica todas as equações de S. O conjunto de todas as substituições unificadoras de S é notado como $U(S)$.

Também podemos definir a solução ótima: a substituição **unificadora mais geral**.

Definição 8.3.1 — Unificadora mais geral. Uma substituição θ é dita **unificadora mais geral** de um sistema de equações S se:

1. O domínio de θ é subconjunto do conjunto de variáveis que ocorrem em S.
2. $\theta \in U(S)$, ou seja, θ é unificadora.
3. θ é mais simples que qualquer outra substituição unificadora γ, ou seja, θ atua em no máximo a mesma quantidade de variáveis que γ.

Além disso, a condição de parada do algoritmo: a **forma resolvida**.

Definição 8.3.2 — Forma resolvida. Uma equação da forma $x = t$ está na **forma resolvida** em um sistema S se x for uma **variável resolvida**: x não ocorre em nenhuma outra equação de S nem em t.

Um sistema de equações está na forma resolvida se todas as suas equações estão.

8.3.1 Regras de Transformação

O algoritmo consiste em aplicar as regras seguintes em alguma equação do sistema até que este esteja na forma resolvida. Se não conseguir, responde com **não** e temos que o sistema não é unificável. Caso haja uma unificação, o método responde com **sim** e devolve a substituição unificadora mais geral.

Eliminação de Equações Triviais $S \cup \{t = t\} \implies S$

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

Eliminação de Variáveis $S \cup \{x = t\} \implies S[t/x] \cup \{x = t\}$, onde x não ocorre em t.

■ **Exemplo 8.5** Determine se $S = \{f(x,g(a,y)), f(x,g(y,x))\}$ é unificável.

Iniciamos com:	$\{f(x,g(a,y)) = f(x,g(y,x))\}$
Decomposição de Termos:	$\{x = x, g(a,y) = g(y,x)\}$
Eliminação de Equações Triviais:	$\{g(a,y) = g(y,x)\}$
Decomposição de Termos:	$\{y = a, x = y\}$
Eliminação de Variáveis ($[a/y]$):	$\{y = a, a = a, x = a\}$
Eliminação de Equações Triviais:	$\{y = a, x = a\}$

S está na forma resolvida: o método encerra respondendo **sim** e nos informa a substituição unificadora mais geral: $[a/y]$. ■

■ **Exemplo 8.6** Determine se $S = \{h(f(a), f(x)), h(f(g(x)), f(g(f(x))))\}$ é unificável.

Iniciamos com:	$\{h(f(a), f(x)) = h(f(g(x)), f(g(f(x))))\}$
Decomposição de Termos:	$\{f(a) = f(g(x)), f(x) = f(g(f(x)))\}$
Decomposição de Termos:	$\{a = g(x), f(x) = f(g(f(x)))\}$

$a = g(x)$ não está na forma resolvida, pois não é da forma $x = t$, e não pode ser eliminada. Assim, o método nos responde **não**. ■

■ **Exemplo 8.7** Determine se $S = \{g(f(y,y)), g(f(h(a),g(b)))\}$ é unificável.

Iniciamos com:	$\{g(f(y,y)) = g(f(h(a),g(b)))\}$
Decomposição de Termos:	$\{f(y,y) = f(h(a),g(b))\}$
Decomposição de Termos:	$\{y = h(a), y = g(b)\}$

$y = h(a)$ não está na forma resolvida, pois y não é uma variável resolvida: y ocorre em outra equação de S. A equação não pode ser eliminada e, portanto, o método nos responde **não**. ■

■ **Exemplo 8.8** Determine se $S = \{h(f(g(x))), h(f(g(g(x)))), h(y)\}$ é unificável.

Iniciamos com:	$\{h(f(g(x))) = h(f(g(g(x))), h(f(g(x))) = h(y), h(f(g(g(x)))) = h(y)\}$
Decomposição de Termos:	$\{f(g(x)) = f(g(g(x))), h(f(g(x))) = h(y), h(f(g(g(x)))) = h(y)\}$
Decomposição de Termos:	$\{g(x) = g(g(x)), h(f(g(x))) = h(y), h(f(g(g(x)))) = h(y)\}$
Decomposição de Termos:	$\{x = g(x), h(f(g(x))) = h(y), h(f(g(g(x)))) = h(y)\}$

$x = g(x)$ não está na forma resolvida, pois x não é uma variável resolvida: x ocorre no termo $g(x)$, o qual está se igualando. A equação não pode ser eliminada e, portanto, o método nos responde **não**. ■

Agora, temos o que precisamos para discutirmos os métodos para se resolver uma instância de SAT.

8.4 Método de Herbrand

Na lógica de primeira ordem, devido a existência de um número infinito de domínios, existe um número infinito de interpretações de uma dada fórmula. Consequentemente, não é possível verificar se uma fórmula é válida ou insatisfatível avaliando-a sob todas as interpretações. Portanto, são necessários procedimentos diferentes para fazer isso.

Vários matemáticos tentaram definir um procedimento geral para a verificação de se uma fórmula é válida ou insatisfatível (por exemplo, Leibniz, Peano e Hilbert), entretanto Church e Turing provaram em 1936 que tal procedimento não existe. Porém, existem procedimentos que verificam se uma fórmula é válida (ou insatisfatível) se ela de fato for válida (ou insatisfatível). Caso não seja, os procedimentos não terminam. Assim, dizemos que a lógica de primeira ordem é **semi-decidível**. Os procedimentos da lógica proposicional sempre terminam, então temos que ela é decidível.

Um importante resultado para provadores automáticos de teoremas foi dado por Herbrand em 1930. Ele desenvolveu um algoritmo que encontra uma interpretação na qual ela não seja válida (ou insatisfatível). Caso ela seja válida (ou insatisfatível), o algoritmo para após um número finito de passos. Caso seja apenas satisfatível ou refutável, o algoritmo não encerra, como esperado.

8.4.1 Universo de Herbrand

Para contornar o problema dos infinitos domínios, Herbrand definiu um domínio H, chamado de **universo de Herbrand**, e provou que basta apenas mostrar que uma fórmula é insatisfatível sob todas as interpretações em H.

Definição 8.4.1 — Universo de Herbrand. Considere uma fórmula φ na forma normal de Skolem e seja S a matriz de φ. O conjunto H é definido pelas seguintes regras:

1. H_0 possui todas as constantes de S. Caso S não possua constantes, H_0 possui uma constante arbitrária a.
2. Para todo símbolo de função n-ária f que ocorre em S e $t_1, ..., t_n$ termos que ocorrem em H_i, $H_{i+1} = H_i \cup \{f(t_1, ..., t_n)\}$

H_∞ é dito **universo de Herbrand**.

Cada H_i é chamado de **conjunto constante**.

8.4.2 Base de Herbrand

Um conceito importante: uma **instância básica** de uma cláusula C de um conjunto S de cláusulas é uma cláusula obtida pela substituição das variáveis de C por elementos do universo de Herbrand de S. Sendo assim, o algoritmo de Herbrand se baseia no seguinte teorema:

Teorema 8.4.1 — Teorema de Herbrand. Um conjunto S de cláusulas é insatisfatível se, e somente se, existe um conjunto finito insatisfatível S' de instâncias básicas de S.

S' é chamado de **base de Herbrand**.

■ **Exemplo 8.9** Mostre que $\phi = \forall x(P(x) \wedge \neg P(f(c)))$ é insatisfatível.

ϕ já está na FNS e seu conjunto de cláusulas é $S = \{P(x), \neg P(f(c))\}$.
Assim, montando o universo de Herbrand de S:

$H_0 = \{c\}$ *(c é uma constante que ocorre em S)*

$H_1 = \{c, f(c)\}$ *(f é um símbolo de função unária que ocorre em S e c é um termo de H_0)*

$H_2 = \{c, f(c), f(f(c))\}$

$H_3 = \{c, f(c), f(f(c)), f(f(f(c)))\}$

...

$H_\infty = \{c, f(c), f(f(c)), f(f(f(c))), f(f(f(f(c)))), ..., f(f(f(f(f(f(f(f(f(f(c))))))))))), ...\}$

E então, para montar a base de Herbrand, usaremos todas as combinações possíveis para substituir as variáveis nas cláusulas com os elementos do universo:

Variáveis	Cláusulas	
x	$P(x)$	$\neg P(f(c))$
c	$P(c)$	$\neg P(f(c))$
$f(c)$	$P(f(c))$	$\neg P(f(c))$

Encontramos $P(f(c))$ e $\neg P(f(c))$: o conjunto é insatisfatível. Assim, ϕ é insatisfatível. ■

■ **Exemplo 8.10** Mostre que $\phi = \forall x((P(x) \rightarrow \exists w Q(w,x)) \wedge P(g(b)) \wedge \neg Q(y,z))$ é insatisfatível.

Colocando ϕ na FNS: $\phi \equiv \forall x((\neg P(x) \vee Q(f(x),x)) \wedge P(g(b)) \wedge \neg Q(y,z))$. Assim, seu conjunto de cláusulas é $S = \{\neg P(x) \vee Q(f(x),x), P(g(b)), \neg Q(y,z)\}$. Montando o universo de Herbrand de S:

$H_0 = \{b\}$

$H_1 = \{b, f(b), g(b)\}$

$H_2 = \{b, f(b), g(b), f(f(b)), f(g(b)), g(f(b)), g(g(b))\}$

$H_3 = H_2 \cup \{f(f(f(b))), f(f(g(b))), f(g(f(b))), f(g(g(b))), g(f(f(b))), g(f(g(b))), g(g(f(b))),$
$g(g(g(b)))\}$

...

$H_\infty = \{b, f(b), g(b), f(f(b)), f(g(b)), ..., g(g(g(g(b))))), ..., f(f(f(f(f(f(f(g(b))))))))))), ...\}$

E então, a base de Herbrand:

Variáveis			Cláusulas		
x	y	z	$\neg P(x) \vee Q(f(x),x)$	$P(g(b))$	$\neg Q(y,z)$
b	b	b	$\neg P(b) \vee Q(f(b),b)$	$P(g(b))$	$\neg Q(b,b)$
b	b	$f(b)$	$\neg P(b) \vee Q(f(b),b)$	$P(g(b))$	$\neg Q(b,f(b))$
b	$f(b)$	b	$\neg P(b) \vee Q(f(b),b)$	$P(g(b))$	$\neg Q(f(b),b)$
			...		
$g(b)$	$f(g(b))$	$g(b)$	$\neg P(g(b)) \vee Q(f(g(b)),g(b))$	$P(g(b))$	$\neg Q(f(g(b)),g(b))$

O algoritmo encerra e temos que S é insatisfatível. Assim, ϕ é insatisfatível. ■

■ **Exemplo 8.11** $\phi = \forall x(R(x) \wedge \neg Q(f(x))$ é insatisfatível?

ϕ já está na FNS e seu conjunto de cláusulas é $S = \{R(x), \neg Q(f(x))\}$.
Assim, montando o universo de Herbrand de S:

$H_0 = \{a\}$ *(S não tem constantes, então H_0 possui uma constante arbitrária)*

$H_1 = \{a, f(a)\}$

$H_2 = \{a, f(a), f(f(a))\}$

$H_3 = \{a, f(a), f(f(a)), f(f(f(a)))\}$

...

$H_\infty = \{a, f(a), f(f(a)), f(f(f(a))), f(f(f(f(a))), ..., f(f(f(f(f(f(f(f(f(f(a))))))))))), ...\}$

E então, a base de Herbrand:

Variáveis	Cláusulas	
x	$R(x)$	$\neg Q(f(x))$
a	$R(a)$	$\neg Q(f(a))$
$f(a)$	$R(f(a))$	$\neg Q(f(f(a)))$
$f(f(a))$	$R(f(f(a)))$	$\neg Q(f(f(f(a))))$
$f(f(f(a)))$	$R(f(f(f(a))))$	$\neg Q(f(f(f(f(a)))))$
	...	
$f(f(f(f(f(f(f(a)))))))$	$R(f(f(f(f(f(f(f(a))))))))$	$\neg Q(f(f(f(f(f(f(f(a))))))))$
	...	

O algoritmo não nos diz nada. Mas cuidado, não podemos afirmar com base na sua indecidibilidade que a fórmula é satisfatível, afinal, ele ainda pode nos afirmar algo no futuro.
Infelizmente, a fórmula é de fato satisfatível, e o algoritmo não terminará. ■

8.4.3 Evolução do método

As tentativas de implementar o método de Herbrand iniciaram em 1960: Paul Gilmore implementou o procedimento pela primeira vez em um computador, entretanto, seu programa se mostrou ineficiente para muitos casos. Davis e Putnam, também em 1960, melhoraram o programa de Gilmore, mas seus resultados também se mostraram ineficientes.

Cinco anos depois, Robinson introduziu o princípio da resolução, que se mostrou muito mais eficiente em relação aos anteriores, e a partir de então, muitos outros procedimentos surgiram. A contribuição de Herbrand abriu portas para prova automática de teoremas que não era possível antes. Ao invés de infinitos diferentes domínios para testar, o domínio era confinado pelo universo de Herbrand, e essa foi a base da Resolução.

8.5 Método da Resolução

O método de Robinson para a lógica de primeira ordem possui os mesmos princípios que o para a lógica proposicional: os conceitos vistos lá, como cláusulas, cláusulas de Horn e a regra da Resolução se mantêm. Além disso, o método continua a aceitar fórmulas somente na forma normal conjuntiva (nesse caso, a matriz está na FNC e a fórmula na FNS). Porém, diferentemente do para a lógica proposicional, introduziremos a unificação de termos para literais complementares.

Esta é a versão da lógica de primeira ordem para a regra da Resolução (t, r, w e s são termos quaisquer e θ é uma substituição unificadora):

$$(P(t_1,...,t_n) \vee Q(r_1,...,r_k)) \wedge (R(w_1,...,w_m) \vee \neg P(s_1,...,s_n)) \equiv$$
$$(P(t_1,...,t_n) \vee Q(r_1,...,r_k)) \wedge (R(w_1,...,w_m) \vee \neg P(s_1,...,s_n)) \wedge \theta((Q(r_1,...,r_k) \vee R(w_1,...,w_m))),$$
$$\text{se } t_1 = s_1,...,t_n = s_n$$

■ **Exemplo 8.12** Prove que $\{\forall x(P(x) \to \exists y Q(x,y)), \forall x \forall y \forall z(Q(x,z) \to P(f(y)))\} \models \forall x(P(x) \to \exists y P(f(y)))$.

Temos que $\Gamma \models \phi$ se, e somente se, $\Gamma \cup \{\neg\phi\}$ é insatisfatível. Assim:
$\Gamma \cup \{\neg\phi\} = \{\forall x(P(x) \to \exists y Q(x,y)), \forall x \forall y \forall z(Q(x,z) \to P(f(y))), \neg\forall x(P(x) \to \exists y P(f(y)))\}$

- $\forall x(P(x) \to \exists y Q(x,y))$:
  $\equiv \forall x(\neg P(x) \vee \exists y Q(x,y))$
  $\equiv \forall x \exists y(\neg P(x) \vee Q(x,y))$ *(Forma Normal Prenex)*
  Adicionando $g(-)$, uma função de Skolem:
  $\equiv \forall x(\neg P(x) \vee Q(x,g(x)))$ *(Forma Normal de Skolem)*
  Matriz: $\neg P(x) \vee Q(x,g(x))$ *(Forma Normal Conjuntiva)*

- $\forall x \forall y \forall z(Q(x,z) \to P(f(y)))$:
  Matriz: $Q(x,z) \to P(f(y))$
  $\equiv \neg Q(x,z) \vee P(f(y))$ *(Forma Normal Conjuntiva)*

- $\neg\forall x(P(x) \to \exists y P(f(y)))$:
  $\equiv \exists x \neg(P(x) \to \exists y P(f(y)))$
  $\equiv \exists x \neg(\neg P(x) \vee \exists y P(f(y)))$
  $\equiv \exists x \exists y \neg(\neg P(x) \vee P(f(y)))$ *(Forma Normal Prenex)*
  Adicionando a e b, duas constantes de Skolem:
  $\equiv \neg(\neg P(a) \vee P(f(b)))$ *(Forma Normal de Skolem)*
  Matriz: $\neg(\neg P(a) \vee P(f(b)))$
  $\equiv P(a) \wedge \neg P(f(b))$ *(Forma Normal Conjuntiva)*

Assim, temos o conjunto de cláusulas $C = \{\neg P(x) \vee Q(x,g(x)), \neg Q(x,z) \vee P(f(y)), P(a), \neg P(f(b))\}$. Vamos aplicar a regra da Resolução.

Suponha C_k para a k-ésima cláusula.

Tomando C_3 e C_4 *(atenção)*: tomemos o sistema $S = \{a = f(b)\}$.
Iniciamos com: $\{a = f(b)\}$.
A equação não está na forma resolvida e não pode ser eliminada. Assim, a unificação não é possível, mas **o método não termina**! Apenas não podemos aplicar a regra nessas cláusulas.

Tomando C_4 e C_1: tomemos o sistema $S = \{f(b) = x\}$.
Iniciamos com: $\{x = f(b)\}$ - *Forma resolvida.*
Com a substituição $[f(b)/x]$, temos $Q(f(b), g(f(b)))$ - C_5;

Tomando C_4 e C_2: tomemos o sistema $S = \{f(b), f(y)\}$.
Iniciamos com: $\{f(y) = f(b)\}$.
Decomposição de Termos: $\{y = b\}$ - *Forma resolvida.*
Com a substituição $[b/y]$, temos $\neg Q(x,z)$ - C_6;

Tomando C_5 e C_6: tomemos o sistema $S = \{x = f(b), z = g(f(b))\}$.
Iniciamos com: $\{x = f(b), z = g(f(b))\}$ - *Forma resolvida.*
Com a substituição $[f(b)/x, g(f(b))/z]$, temos $()$;

Com o surgimento da cláusula vazia, temos que $\Gamma \cup \{\neg\phi\}$ é insatisfatível. Logo, $\Gamma \models \phi$. ∎

∎ **Exemplo 8.13** Prove que, se uma relação é de equivalência, então ela é circular.

Reflexividade $\forall x(R(x,x))$

Simetria $\forall x \forall y(R(x,y) \rightarrow R(y,x))$

Transitividade $\forall x \forall y \forall z((R(x,y) \wedge R(y,z)) \rightarrow R(x,z))$

Circular $\forall x \forall y \forall z((R(x,y) \wedge R(y,z)) \rightarrow R(z,x))$

Queremos provar que $\{\forall x(R(x,x)), \forall x \forall y(R(x,y) \rightarrow R(y,x)), \forall x \forall y \forall z((R(x,y) \wedge R(y,z)) \rightarrow R(x,z))\} \vdash \forall x \forall y \forall z((R(x,y) \wedge R(y,z)) \rightarrow R(z,x))$. Lembramos que $\Gamma \vdash \phi \leftrightarrow \Gamma \models \phi$. Assim:

$\Gamma \cup \{\neg\phi\} = \{\forall x(R(x,x)), \forall x \forall y(R(x,y) \rightarrow R(y,x)), \forall x \forall y \forall z((R(x,y) \wedge R(y,z)) \rightarrow R(x,z)),$
$\neg(\forall x \forall y \forall z((R(x,y) \wedge R(y,z)) \rightarrow R(z,x)))\}$

- $\forall x(R(x,x))$:
  Matriz: $R(x,x)$ *(Forma Normal Conjuntiva)*

- $\forall x \forall y(R(x,y) \rightarrow R(y,x))$:
  Matriz: $R(x,y) \rightarrow R(y,z)$
  $\equiv \neg R(x,y) \vee R(y,z)$ *(Forma Normal Conjuntiva)*

- $\forall x \forall y \forall z((R(x,y) \wedge R(y,z)) \rightarrow R(x,z))$:
  Matriz: $(R(x,y) \wedge R(y,z)) \rightarrow R(x,z)$
  $\equiv \neg(R(x,y) \wedge R(y,z)) \vee R(x,z)$
  $\equiv \neg R(x,y) \vee \neg R(y,z) \vee R(x,z)$ *(Forma Normal Conjuntiva)*

- $\neg(\forall x \forall y \forall z((R(x,y) \wedge R(y,z)) \rightarrow R(z,x)))$:
  $\equiv \exists x \exists y \exists z \neg((R(x,y) \wedge R(y,z)) \rightarrow R(z,x))$ *(Forma Normal Prenex)*
  Adicionando a, b e c, três constantes de Skolem:
  $\equiv \neg((R(a,b) \wedge R(b,c)) \rightarrow R(c,a))$ *(Forma Normal de Skolem)*
  Matriz: $\neg((R(a,b) \wedge R(b,c)) \rightarrow R(c,a))$
  $\equiv \neg(\neg(R(a,b) \wedge R(b,c)) \vee R(c,a))$
  $\equiv \neg\neg(R(a,b) \wedge R(b,c)) \wedge \neg R(c,a)$
  $\equiv R(a,b) \wedge R(b,c) \wedge \neg R(c,a)$ *(Forma Normal Conjuntiva)*

Assim, temos o conjunto de cláusulas
$C = \{\neg R(x,y) \vee R(y,z), \neg R(x,y) \vee \neg R(y,z) \vee R(x,z), R(a,b), R(b,c), \neg R(c,a)\}$

Suponha C_k para a k-ésima cláusula.

Tomando C_3 e C_1: tomemos o sistema $S = \{x = a, y = b\}$.
Iniciamos com: $\{x = a, y = b\}$ - *Forma resolvida.*
Com a substituição $[a/x, b/y]$, temos $R(b,z)$ - C_6;

Tomando C_1 e C_5: tomemos o sistema $S = \{y = c, z = a\}$.
Iniciamos com: $\{y = c, z = a\}$ - *Forma resolvida.*
Com a substituição $[c/y, a/z]$, temos $\neg R(x,c)$ - C_7;

Tomando C_6 e C_7: tomemos o sistema $S = \{x = b, z = c\}$.
Iniciamos com: $\{x = b, z = c\}$ - *Forma resolvida.*
Com a substituição $[b/x, c/z]$, temos $()$;
Com o surgimento da cláusula vazia, temos que $\Gamma \cup \{\neg\phi\}$ é insatisfatível. Logo, $\Gamma \models \phi$ e a prova está completa. ∎

9. Limites da Lógica Simbólica

Vimos até agora as grandes potencialidades da lógica simbólica na resolução de validade de argumentos, consistência de um conjunto de sentenças, satisfatibilidade etc. Vamos fazer uma reflexão sobre os limites dessa abordagem. Ela nos levará a um resultado um tanto quanto intrigante, ao qual chegou o matemático austríaco Kurt Gödel que mostra que, na Aritmética, nem tudo é verdadeiro. Isso revela uma faceta um tanto misteriosa da Lógica: nem todas as verdades matemáticas podem ser provadas. A motivação de Gödel partiu de um programa de pesquisa bastante ambicioso liderado pelo matemático alemão David Hilbert em cerca do final do século XIX.

9.1 O Programa de Hilbert

De forma a ganhar confiança na solidez das teorias da matemática e das ciências exatas (incluindo suas grandes áreas, como Geometria, Aritmética, Álgebra...), no sentido de que essas teorias estariam livres de contradições, Hilbert buscou aplicar o método da lógica simbólica para mostrar que o conjunto de leis formalizadas como sentenças da lógica de primeira ordem é consistente (satisfatível).

Se isso pudesse ser feito de forma definitiva, as teorias seriam corretas (permitiriam provar apenas o que fosse verdadeiro) e completas (não haveria sentença verdadeira que não foi demonstrada) com relação às estruturas matemáticas que pretendiam formalizar.

Em 1889, por exemplo, o matemático italiano Giuseppe Peano publicava um artigo propondo uma teoria formal da Aritmética consistindo de 7 leis básicas, formalizadas na assinatura da Aritmética:

1. $\neg\exists x(a = s(x))$ - O 0 não é sucessor de nenhum número.
2. $\forall x\forall y((s(x) = s(y)) \rightarrow (x = y))$ - A função sucessor é injetora.
3. $(P(a) \wedge \forall n(P(n) \rightarrow P(s(n)))) \rightarrow \forall x P(x)$ - Lei da Indução Matemática.
4. $\forall x\forall y((x + s(y)) = (s(x + y)))$ - Recursividade da Adição.

5. $\forall x((x+a)=x)$ - 0 é elemento neutro na adição.
6. $\forall x\forall y((x\times s(y))=((x\times y)+x)$ - Recursividade da Multiplicação.
7. $\forall x((x\times a)=a)$ - Qualquer número multiplicado a 0 resulta em 0.

9.2 O Teorema da Incompletude

O trabalho de Peano parecia ser promissor. Contudo, Gödel mostrou que qualquer teoria que se proponha a formalizar a Aritmética **não poderia ser correta e completa ao mesmo tempo**.

9.2.1 A estratégia de Gödel

"Eu não sou verdadeira.": essa sentença é um paradoxo — mais especificamente, o **Paradoxo do Mentiroso** —, pois é uma afirmação impossível de determinar seu valor-verdade: ela é verdadeira se, e somente se, não for verdadeira. Gödel tomou uma variante do paradoxo:

"Eu não sou demonstrável."

E tentou formalizá-la na assinatura da Aritmética. Para isso, estabeleceu um correspondência entre sentenças e números naturais, tomando por base o **Teorema Fundamental da Aritmética**, que diz que todo inteiro possui uma fatoração prima e ela é única. Assim, atribuiu a cada símbolo da assinatura (não lógicos) e aos símbolos lógicos um natural primo distinto, e calculou o número natural correspondendo à sentença, fazendo a multiplicação de potências de primos:

$\forall$	$\exists$	$\rightarrow$	$=$	$($	$)$	$\neg$	$+(-,-)$	$\wedge$	$\times(-,-)$	x	y	a	$s(-)$
2	3	5	7	11	13	17	19	23	29	31	37	43	45

E assim, ele poderia calcular um número natural correspondente a uma sentença formalizada da Aritmética usando o TFA. Por exemplo,

$$\neg\exists x(a=s(x))=$$
$$2^{17+1}\times 3^{3+1}\times 5^{31+1}\times 7^{11+1}\times 11^{43+1}\times 13^{7+1}\times 15^{45+1}\times 17^{11+1}\times 17^{31+1}\times 19^{13+1}\times 23^{13+1}$$

Onde os expoentes - 1 são os números correspondentes aos símbolos.

Em razão disso, o predicado Demonstrável$(-)$ agora se torna um predicado entre números, e ele o definiu usando a linguagem da Aritmética. Desse modo, o paradoxo *eu sou verdadeira se, e somente se, eu não sou demonstrável* pode ser escrito na linguagem da Aritmética, provando o seguinte teorema:

Teorema 9.2.1 — Teorema da Incompletude. Qualquer teoria que se proponha a formalizar a Aritmética na lógica de primeira ordem não pode ser correta e completa ao mesmo tempo.

O teorema também pode ser chamado de Teorema da Incorretude.