Lógica Computacional

José Manuel E. Valença *

31 de Maio de 2006

*Departamento de Informática, Universidade do Minho, Campus de Gualtar Braga

1.Lógica Proposicional

Genericamente a **Lógica** associa a **frases** de uma linguagem o atributo de **validade**.

Frases de uma lógica, construídas através de regras formais de sintaxe, são designadas por **fórmulas** e formam sempre um conjunto enumerável construído indutivamente.

Neste curso iremos apresentar duas visões essenciais da validade de fórmulas:

Validade Clássica: a cada proposição está associado um e só um de dois **valores de verdade** (denotados pelos símbolos $\mathbf{1}$ e $\mathbf{0}$ ou `verdadeiro` e `falso`); a proposição é válida se lhe é associado o valor $\mathbf{1}$ ou `verdadeiro`.

Validade Intuicionista: uma proposição é válida se e só se é possível construir uma **prova** dessa validade.

Exemplo

chove amanhã ou não chove amanhã

é representada por uma proposição $\mathbf{p} \vee (\neg \mathbf{p})$ que é sempre válida na abordagem clássica mas não é necessariamente válida na abordagem intuicionista.

De facto intuicionisticamente a frase só seria válida se fosse possível provar que amanhã chove ou então se fosse possível provar que amanhã não chove.

1.1. Sintaxe

Frases que, num determinado contexto, são **logicamente atómicas** no sentido em que a sua validade é independente da validade de qualquer outra frase são representadas por **símbolos proposicionais**. Numa **Lógica Proposicional**, todas as frases que não são representadas por esses símbolos são construídas por combinação de outras frases usando **conectivos**.

Diferentes conjuntos de conectivos definem diferentes **linguagens proposicionais**. A mais "comum", designada por **Lógica Proposicional Clássica**

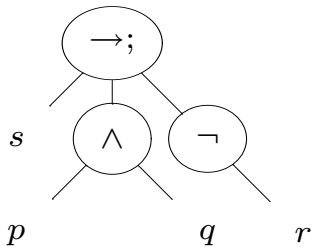
1 DEFINIÇÃO *Dado um conjunto de símbolos proposicionais $\mathcal{P}$ a linguagem proposicional clássica $\mathcal{L}_{\mathcal{P}}$ é o menor conjunto de frases que contém $\mathcal{P}$ e ainda*

$$\bot\ ,\ \neg\phi\ ,\ (\phi)\ ,\ \phi\wedge\varphi\ ,\ \phi\vee\varphi\ ,\ \phi\supset\varphi\ ,\ \eta\longrightarrow\phi;\varphi \qquad (1)$$

para todo $\phi,\varphi,\eta\in\mathcal{L}_{\mathcal{P}}$.

Note-se que . . .

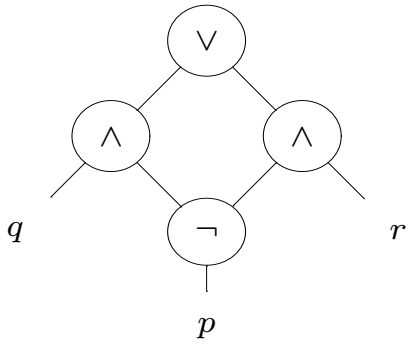
Um objecto assim gerado tem, naturalmente, uma representação arbórea. Por exemplo, a fórmula $s\longrightarrow(p\wedge q);\neg r$ é representado pela árvore



Numa perspectiva de representação de dados, estas árvores transformam-se naturalmente em **grafos acíclicos** quando se introduz a possibilidade de partilha de sub-fórmulas.

Considere-se, por exemplo, a fórmula $(q\wedge\neg p)\vee(\neg p\wedge r)$ onde a sub-fórmula $\neg p$

ocorre duas vezes. O grafo seguinte usa essa duplicação para simplificar a representação



Várias variantes desta linguagem proposicional podem ser definidas.

Exemplos

Lógica Hilbertiana mínima: $(\mathcal{H}_{\mathcal{P}})$

Linguagem definida pelos conectivos $\{\bot, \supset\}$; a conversão de fórmulas em $\mathcal{L}_{\mathcal{P}}$ em fórmulas de $\mathcal{H}_{\mathcal{P}}$ obtém-se por aplicação sucessiva das seguintes regras de reescrita

$\phi \vee \varphi$	$\leadsto (\neg\phi) \supset \varphi$	$\phi \wedge \varphi$	$\leadsto \neg(\neg\phi \vee \neg\varphi)$
$\neg\phi$	$\leadsto \phi \supset \bot$	$\eta \rightarrow \phi;\varphi$	$\leadsto (\eta \wedge \phi) \vee (\neg\eta \wedge \varphi)$

Lógica de Formas Normais Negativas: $(\mathcal{L}_{\mathcal{P}n})$

Lógica definida pelos conectivos $\neg, \vee, \wedge$, com a restrição de a negação $\neg$ ser aplicado, apenas, a símbolos proposicionais; portanto todas as fórmulas têm uma das quatro formas seguintes

$$p \quad , \quad \neg p \quad , \quad \phi \wedge \varphi \quad , \quad \phi \vee \varphi \qquad \text{com} \quad p \in \mathcal{P} \quad \phi, \varphi \in \mathcal{L}_{\mathcal{P}n}$$

A conversão de $\mathcal{L}_{\mathcal{P}}$ em $\mathcal{LP}_n$ é feita por aplicação sucessiva das seguintes regras de reescrita:

$(\phi \supset \varphi)$	$\leadsto (\neg\phi) \vee \varphi$	$(\eta \rightarrow \phi;\varphi)$	$\leadsto (\eta \wedge \phi) \vee (\neg\eta \wedge \varphi)$
$\neg(\neg p)$	$\leadsto p$	$\bot$	$\leadsto p \wedge \neg p \quad p \in \mathcal{P}$
$\neg(\phi \wedge \varphi)$	$\leadsto \neg\phi \vee \neg\varphi$	$\neg(\phi \vee \varphi)$	$\leadsto \neg\phi \wedge \neg\varphi$

Grafos de Shannon $(\mathcal{S}_{\mathcal{P}})$

Linguagem definida pelos conectivos $\{\top, \bot, (\cdot \rightarrow \cdot; \cdot)\}$ com a restrição de que o primeiro argumento do conectivo $(\cdot \rightarrow \cdot; \cdot)$ ser sempre um símbolo proposicional. Assim são apenas três as formas possíveis de fórmulas desta linguagem:

$$\top \quad , \quad \bot \quad , \quad (p \rightarrow \phi; \varphi) \qquad \text{com} \quad p \in \mathcal{P} \quad \phi, \varphi \in \mathcal{S}_{\mathcal{P}}$$

A conversão de $\mathcal{L}_{\mathcal{P}}$ em $\mathcal{S}_{\mathcal{P}}$ passa pelo passo intermédio de conversão em formas normalizadas. A conversão de $\mathcal{L}_{\mathcal{P}_n}$ em $\mathcal{S}_{\mathcal{P}}$ obtém-se por aplicação sucessiva das seguintes regras

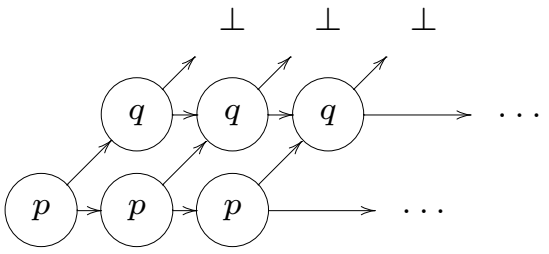
$p \rightsquigarrow (p \rightarrow \top; \bot)$ $\neg p \rightsquigarrow (p \rightarrow \bot; \top)$	$(p \rightarrow \phi; \varphi) \vee \eta \rightsquigarrow (p \rightarrow \phi \vee \eta; \varphi \vee \eta)$ $(p \rightarrow \phi; \varphi) \wedge \eta \rightsquigarrow (p \rightarrow \phi \wedge \eta; \varphi \wedge \eta)$

Parece claro que os grafos de Shannon assim produzidos são excessivamente complexos e podem ser simplificados. Por exemplo é possível usar directamente regras de simplificação como

$$\phi \vee \top \rightsquigarrow \top \qquad \phi \wedge \bot \rightsquigarrow \bot \qquad \phi \vee \bot \rightsquigarrow \phi \qquad \phi \wedge \top \rightsquigarrow \phi$$

De facto a grande vantagem da representação de fórmulas proposicionais em grafos de Shannon está na facilidade com que se definem e implementam regras de simplificação muito poderosas que permitem, de forma muito eficiente, fazer juízos sobre a validade destas fórmulas.

A representação, por grafo acíclico de uma fórmula $\phi \in \mathcal{S}_{\mathcal{P}}$ é feita usando regras específicas: as folhas são sempre marcadas com um dos símbolos $\top, \bot$ e os nodos (que não folhas) são marcados por símbolos proposicionais.



Representa uma "pseudo-fórmula" infinita ϕ gerada pelas seguintes equações

$$\phi \doteq p \rightarrow \varphi; \phi$$
$$\varphi \doteq q \rightarrow \bot; \varphi$$

1.2. Linguagens de Clausulas

No contexto da Lógica Proposicional, um **literal** é um símbolo proposicional $p \in \mathcal{P}$ ou a sua negação $\neg p$.

As **linguagens de clausulas** são duas formas particulares da linguagem das formas normais negativas assim definidas:

Formas Normais Conjuntivas ($\mathcal{L}_{\mathcal{P}_{cn}}$) sub-linguagem de $\mathcal{L}_{\mathcal{P}_n}$ formada por todas as fórmulas onde nenhuma conjunção aparece como argumento de uma disjunção.

As fórmulas nesta linguagem são sempre conjunções de disjunções de literais; i.e. a sua forma mais geral é

$$(a_1 \vee \ldots \vee a_k) \wedge (b_1 \vee \ldots \vee b_l) \wedge \ldots \wedge (c_1 \vee \ldots \vee c_n)$$

em que os $a_i, b_i, \ldots, c_i$ são literais.

Formas Normais Disjuntivas ($\mathcal{L}_{\mathcal{P}_{dn}}$) sub-linguagem de $\mathcal{L}_{\mathcal{P}_n}$ formada por todas as fórmulas onde nenhuma disjunção aparece como argumento de uma conjunção.

As fórmulas nesta linguagem são sempre disjunções de conjunções de literais

$$(a_1 \wedge \ldots \wedge a_k) \vee (b_1 \wedge \ldots \wedge b_l) \vee \ldots \vee (c_1 \wedge \ldots \wedge c_n)$$

Em qualquer um destes casos uma fórmula pode ser representada sempre por um conjunto de conjuntos de literais.

Tomemos por referência o caso das formas normais conjuntivas (o outro caso é perfeitamente dual). Uma fórmula ϕ pode ser representada por

$$\phi = C_1 \wedge C_2 \wedge \ldots \wedge C_n$$

$$\text{com} \quad C_i \doteq a_1^i \vee a_2^i \vee \ldots \vee a_n^i$$

As fórmulas C_i designam-se por **clausulas**.

Cada claúsula pode ser representada por um conjunto de literais

$$C = \{a_1 \ , \ a_2 \ , \ \ldots \ , \ a_n\}$$

e a fórmula ϕ pode ser representada pelo conjunto das clausulas

$$\phi = \{C_1 \ , \ C_2 \ , \ \ldots \ , \ C_k\}$$

Esta representação de fórmulas é válida tanto para as formas conjuntivas como para as formas disjuntivas. A diferença está em que, no primeiro caso, as cláusula C_i têm implícita a disjunção de todos os seus elementos e a fórmula ϕ tem implícita a conjunção das claúsulas. No caso das formas disjuntivas, em cada cláusula C_i está implícita a conjunção dos seus literais e na fórmula ϕ está implícita a disjunção das cláusulas.

exemplo:
Considere-se a fórmula seguinte numa das duas linguagens de cláusulas

$$\{\{p, \neg q, r\} \ , \ \{\neg p, \neg r\} \ , \ \{q, \neg r\}\}$$

Em $\mathcal{L}_{\mathcal{P}_{cn}}$ esta seria a representação da fórmula

$$(p \vee \neg q \vee r) \wedge (\neg p \vee \neg r) \wedge (q \vee \neg r)$$

Em $\mathcal{L}_{\mathcal{P}_{dn}}$ seria a representação da fórmula

$$(p \wedge \neg q \wedge r) \vee (\neg p \wedge \neg r) \vee (q \wedge \neg r)$$

□

Esta representação vai permitir que a manipulação das fórmulas (para provar a sua validade, por exemplo) se resuma a uma transformação de conjuntos; fica assim diluída toda a complexidade inerente à existência de conectivos diversos com interpretações diversas. Com esta estrutura basta definir as regras de manipulação de conjuntos que tenham o significado lógico apropriado.

A construção de formas normais conjuntivas (ou disjuntivas) a partir de formas normais negativas é o nosso próximo objectivo.

Antes, porém, é preciso sistematizar o tipo de operações genéricas que são admissíveis nos "conjuntos de conjuntos de literais". Vamos lidar, essencialmente, com duas operações:

soma A soma de dois destas conjuntos, μ e υ, representada por $\mu + \upsilon$, é a união de conjuntos (com eventuais repetições). Por exemplo,

$$\{\{a,b\}\ ,\ \{c,d\}\} + \{\{x,y\}\ ,\ \{z,w\}\} = \\ = \{\{a,b\}\ ,\ \{c,d\}\ ,\ \{x,y\}\ ,\ \{z,w\}\}$$

produto O produto $\mu \times \upsilon$ é o conjunto formada por todas as uniões de elementos de μ com elementos de υ. Por exemplo

$$\{\{a,b\}\ ,\ \{c,d\}\} \times \{\{x,y\}\ ,\ \{z,w\}\} = \\ = \{\{a,b,x,y\}\ ,\ \{c,d,x,y\}\ ,\ \{a,b,z,w\}\ ,\ \{c,d,z,w\}\}$$

exemplo:
Considere-se duas proposições em forma normal conjuntiva $\mu = (a \wedge b)$ e $\upsilon = (c \wedge d)$, sendo a, b, c, d literais; como conjuntos teríamos

$$\mu = \{\{a\}\ ,\ \{b\}\} \quad , \quad \upsilon = \{\{c\}\ ,\ \{d\}\}$$

Da associatividade e distributividade de $\wedge, \vee$ conclui-se

$$\mu \wedge \upsilon = a \wedge b \wedge c \wedge d$$
$$\mu \vee \upsilon = (a \vee c) \wedge (a \vee d) \wedge (b \vee c) \wedge (b \vee d)$$

Como conjuntos teríamos

$$\mu \wedge \upsilon = \{\{a\}\ ,\ \{b\}\ ,\ \{c\}\ ,\ \{d\}\} = \mu + \upsilon$$
$$\mu \vee \upsilon = \{\{a,c\}\ ,\ \{a,d\}\ ,\ \{b,c\}\ ,\ \{b,d\}\} = \mu \times \upsilon$$

Este exemplo sugere a justificação da seguinte proposição

1 PROPOSIÇÃO *Se* μ *e* υ *são duas formas conjuntivas representadas por conjuntos de conjuntos de literais então* $\mu \wedge \upsilon$ *é representado pela conjunto* $\mu + \upsilon$ *e* $\mu \vee \upsilon$ *é representado pela conjunto* $\mu \times \upsilon$.

Trocando os papeis dos símbolos $\wedge$ e $\vee$ conclui-se também

2 PROPOSIÇÃO *Se* μ *e* υ *são duas formas disjuntivas representadas por conjuntos de conjuntos de literais então* $\mu \vee \upsilon$ *é representado pelo conjunto* $\mu + \upsilon$ *e* $\mu \wedge \upsilon$ *é representado pelo conjunto* $\mu \times \upsilon$.

Nestas linguagens têm importância especial as *clausulas* e as *formas* vazias.

Numa forma conjuntiva a *clausula vazia* representa uma disjunção $C = \bigvee_{i=1}^{n} a_i$ em que o número de literais a_i é zero. Tal cláusula[1] é equivalente a $\bot$.

Qualquer forma conjuntiva que contenha a clausula vazia tem a forma $\varphi = C \wedge \bot$ e, portanto, é equivalente a $\bot$.

Por dualidade, nas formas disjuntivas, a *clausula vazia* representa uma conjunção $C = \bigwedge_{i=1}^{n} a_i$ em que o número de literais é nulo; ou seja, é uma representação de $\top$.

Qualquer forma disjuntiva que contenha a clausula vazia tem a forma $\varphi = C \vee \top$ e, portanto, é equivalente a $\top$.

Finalmente, uma *forma conjuntiva vazia* (representada por um conjunto vazio de clausulas) é a conjunção de zero elementos e, portanto, é equivalente a $\top$. Dualmente a *forma disjuntiva vazia* é uma disjunção de zero elementos e, assim, é equivalente a $\bot$.

[1] Para ser válida teria de existir algum literal que fosse válido; como não existe nenhum a cláusula será sempre inválida.

1.3. Validade e Verificação

Ao longo desta secção vamos definir a noção de **validade clássica** da lógica proposicional $\mathcal{L}_{\mathcal{P}}$.

Essencialmente vamos definir uma estratégia para associar valores de verdade a fórmulas e analisar como é que se distribuem esses valores de verdade quando examinamos conjuntos (eventualmente infinitos) de fórmulas.

O modo mais directo de analisar conjuntos de fórmulas é através da noção de **esquema de fórmula**. Analisaremos em seguida este conceito, bem como a noção de **substituição** que lhe está associada.

A **atribuição** de valores de verdade a fórmulas requer a noção de **modelo** que se prende com outros conceitos como o de **consequência**. Analisaremos também estas três noções essenciais.

□

Sejam $\mathcal{V} = \{A, B, C, \ldots\}$ e $\mathcal{P} = \{p, q, r, s, \ldots\}$ um conjunto finito de *variáveis* e um conjunto, não necessariamente finito, de *símbolos proposicionais*.

Um **esquema de fórmulas** de $\mathcal{L}_{\mathcal{P}}$ é uma fórmula em $\mathcal{L}_{\mathcal{P}\cup\mathcal{V}}$.

exemplo: Se for $\mathcal{V} = \{A, B, C\}$ então a frases a seguir apresentadas são esquemas de fórmulas em qualquer $\mathcal{L}_{\mathcal{P}}$

$$A \vee \neg A \qquad ((A \supset B) \supset A) \supset A \qquad (A \supset C) \supset (B \supset C) \supset (A \vee B) \supset C$$

Cada um destes objectos representa um conjunto infinito de fórmulas que são geradas substituindo as variáveis por fórmulas concretas na linguagem $\mathcal{L}_{\mathcal{P}}$. Assim, como instâncias de cada um destes esquemas, temos as fórmulas

$$(p \vee q) \vee \neg(p \vee q) \quad , \quad ((p \wedge r \supset \neg q) \supset p \wedge r) \supset p \wedge r \quad ,$$
$$(p \supset p \vee q) \supset (q \supset p \vee q) \supset (p \vee q) \supset (p \vee q)$$

O conjunto das **variáveis livres** $\mathcal{V}_\phi$ de um esquema de fórmulas ϕ é definido recursivamente pelas seguintes regras:

(i) $\mathcal{V}_p = \mathcal{V}_\bot = \emptyset$ para todo $p \in \mathcal{P}$
(ii) $\mathcal{V}_A = \{A\}$ para todo $A \in \mathcal{V}$
(iii) $\mathcal{V}_{\neg\phi} = \mathcal{V}_\phi \quad , \quad \mathcal{V}_{\eta\to\phi;\varphi} = \mathcal{V}_\eta \cup \mathcal{V}_\phi \cup \mathcal{V}_\varphi$
(iv) $\mathcal{V}_{\phi\circ\varphi} = \mathcal{V}_\phi \cup \mathcal{V}_\varphi$ para $\circ \in \{\wedge, \vee, \supset\}$

Um esquema é uma **fórmula de base** quando não contém quaisquer variáveis livres.

Uma **substituição** é uma função $\upsilon : \mathcal{V} \longrightarrow \mathcal{L}_{\mathcal{P}\cup\mathcal{V}}$ que associa variáveis a esquemas de fórmulas de $\mathcal{L}_\mathcal{P}$.

Qualquer substituição υ é extensível a uma função $\bar{\upsilon} : \mathcal{L}_{\mathcal{P}\cup\mathcal{V}} \to \mathcal{L}_{\mathcal{P}\cup\mathcal{V}}$ de esquemas em esquemas definida recursivamente por:

(i) $\bar{\upsilon}(A) = \upsilon(A)$ para todo $A \in \mathcal{V}$
(ii) $\bar{\upsilon}(\phi) = \phi$ sempre que ϕ é uma fórmula de base,
(iii) $\bar{\upsilon}(\neg\phi) = \neg\bar{\upsilon}(\phi)$ e $\bar{\upsilon}(\eta \to \phi; \varphi) = \bar{\upsilon}(\eta) \to \bar{\upsilon}(\phi); \bar{\upsilon}(\varphi)$
(iv) $\bar{\upsilon}(\phi \circ \varphi) = \bar{\upsilon}(\phi) \circ \bar{\upsilon}(\varphi)$ para $\circ \in \{\wedge, \vee, \supset\}$

A substituição que associa a variável A ao esquema Φ e deixa todas as restantes variáveis imutáveis é representada por $[\Phi/A]$. A sua aplicação a um esquema φ é representada por $\varphi[\Phi/A]$.

Genericamente representa-se por

$$[\Phi_1/A_1, \Phi_2/A_2, \ldots, \Phi_n/A_n]$$

a substituição que associa cada uma das variáveis A_i ao esquema Φ_i e deixa inalteráveis as restantes variáveis.

□

Em $\mathcal{L}_{\mathcal{P}}$ um **modelo** é um qualquer subconjunto de $\mathcal{P}$.

Essencialmente um modelo $\mathcal{M} \subseteq \mathcal{P}$ estabelece quais os símbolos proposicionais a que deve ser atribuído o valor de verdade $\mathbf{1}$.

Vamos representar o conjunto dos valores de verdade por $\mathbb{Z}_2$. Algebricamente é um corpo finito com o conjunto de suporte $\{0, 1\}$ e duas operações binárias associativas e comutativas $+$ e $*$ que são definidos pelas relações

$$x + x = 0, \quad x + 0 = x, \quad x + 1 \neq x, \quad x * x = x, \quad x * 1 = x$$

Note-se que . . .

numa lógica booleana, $+$ é a operação de `xor` ("ou" exclusivo) enquanto que $*$ é a operação de `and`.

A **atribuição** de valores de verdade definida pelo modelo $\mathcal{M}$ é uma função $\mu : \mathcal{L}_{\mathcal{P}} \to \mathbb{Z}_2$ definida recursivamente por

(i) $\mu(p) = 1$ se $p \in \mathcal{M}$ e $\mu(p) = 0$ se $p \notin \mathcal{M}$
(ii) $\mu(\neg\phi) = 1 + \mu(\phi) \quad , \quad \mu(\bot) = 0$
(iii) $\mu(\phi \wedge \varphi) = \mu(\phi) * \mu(\varphi)$
(iv) $\mu(\phi \vee \varphi) = \mu(\phi) + \mu(\varphi) + \mu(\phi) * \mu(\varphi)$
(v) $\mu(\phi \supset \varphi) = 1 + \mu(\phi) + \mu(\phi) * \mu(\varphi)$
(vi) $\mu(\eta \to \phi; \varphi) = \mu(\eta) * \mu(\phi) + (1 + \mu(\eta)) * \mu(\varphi)$

2 DEFINIÇÃO *Uma fórmula ϕ **é válida** no modelo $\mathcal{M}$, representado por $\mathcal{M} \models \phi$, se e só se $\mu(\phi) = 1$.*

*ϕ é uma **tautologia**, representado por $\models \phi$, se é válida em <u>todos</u> os modelos $\mathcal{M} \subseteq \mathcal{P}$.*

ϕ é ***verificável*** *se existe pelo menos um modelo $\mathcal{M}$ onde a fórmula é válida (ou, equivalentemente, quando $\neg\phi$ não for tautologia).*

Estes conceitos podem ser estendidos para conjuntos finitos de fórmulas $\Gamma = \{\phi_1, \phi_2, \ldots\}$, designados por **teorias**.

No que se segue representamos por $\neg\Gamma$ a teoria $\{\neg\phi_1, \neg\phi_2, \ldots\}$.

3 DEFINIÇÃO *Um conjunto de fórmulas Γ é* ***inconsistente****, representado por $\Gamma \models$, quando nenhum modelo $\mathcal{M}$ consegue validar, simultaneamente, todas as fórmulas de Γ .*

A fórmula ϕ é uma ***consequência*** *de Γ , representado por $\Gamma \models \phi$, quando $\Gamma \cup \{\neg\phi\}$ é inconsistente.*

Γ é ***refutado*** *quando $\neg\Gamma$ não é inconsistente.*

Frases como

> *" ϕ é válida em $\mathcal{M}$ ", " ϕ é tautologia ", " ϕ é verificável ", " ϕ é consequência de Γ ", " Γ é inconsistente " e " Γ é refutado ",*

são asserções na linguagem que pretende especificar a lógica que se está a definir. São frases de uma linguagem cuja função é descrever outra linguagem; neste caso as frases permitem descrever a lógica proposicional.

Nestas situações é costume designar a lógica que está a ser descrita por **lógica objecto** e a linguagem lógica que está a ser usada para a descrever por **meta-lógica**.

As frases que ilustramos são fórmulas na meta-lógica a que se dá o nome de **juízo**. Estes juízos possuem uma representação simbólica própria; respectivamente

$$\mathcal{M} \models \phi, \quad \models \phi, \quad \not\models \neg\phi, \quad \Gamma \models \phi, \quad \Gamma \models, \quad \neg\Gamma \not\models$$

e servem para fazer asserções básicas sobre o significado destas várias componentes da lógica. Como qualquer frase lógica, os juízos podem ser válidos ou não.

Temos de ter em atenção que tais juízos podem também ser feitos usando esquemas de fórmulas em vez de fórmulas[2]. Tomemos por exemplo a frase

$$\models A \vee \neg A$$

Agora temos um juízo que depende do valor da variável A; é um juízo com uma variável livre. Substituindo A por fórmulas de base constrói-se uma infinidade de juízos base; se quisermos afirmar que todos esses juízos base são válidos é necessário um juízo que quantifique em relação a todos os eventuais valores de A.

Representaremos tal juízo pela *quantificação*

$$\bigwedge A. \quad \models A \vee \neg A$$

Os juízos podem ser combinados por conectivos tal como as fórmulas da lógica proposicional. Para isso temos de introduzir alguns novos símbolos de conectivas com o cuidado de os distinguir dos que são usados na lógica-objecto.

Em primeiro lugar, a *negação*. Na meta-lógica a negação aparece como a barra / combinada com o operador $\models$. Assim o juízo $\not\models \phi$ é a negação do juízo $\models \phi$.

Para a *conjunção* de dois juízos usa-se o símbolo $\&$. Por exemplo, para escrever *"ϕ é tautologia e φ é sua consequência"*, escreveríamos

$$(\models \phi) \,\&\, (\phi \models \varphi)$$

A *implicação* ao meta-nível é representada por $\Longrightarrow$. Por exemplo, a frase: *"qualquer que sejam as fórmulas A e B, se A e $A \supset B$ são tautologia,*

[2] Aliás é esta a razão principal para se ter introduzido a noção de esquema.

então B *é tautologia*", é representada pelo juízo

$$\bigwedge A, B.\quad (\models A)\quad \&\quad (\models A \supset B)\quad \Longrightarrow\quad (\models B)$$

onde convergem todas as componentes da nossa linguagem de juízos.

1.4. Redução Semântica

Duas fórmulas ϕ e φ são **semanticamente equivalentes**, representado por $\phi \equiv \varphi$, se são válidas exactamente nos mesmos modelos; i.e.,

$$\mathcal{M} \models \phi \iff \mathcal{M} \models \varphi \qquad \text{para todo modelo} \quad \mathcal{M} \subseteq \mathcal{P}$$

Como propriedade essencial da equivalência semântica tem-se

3 PROPOSIÇÃO *Se $J(\phi)$ é um juízo semântico onde ocorre a fórmula ϕ, e se for $\phi \equiv \varphi$, então a validade desse juízo é precisamente a mesma da do juízo $J(\varphi)$ que se obtém do primeiro substituindo todas as ocorrências de ϕ por φ.*

O objectivo da **redução semântica** é a substituição de uma fórmula por outra que lhe seja semanticamente equivalente mas que seja, de alguma forma, mais simples.

exemplo: As duas seguintes fórmulas (a primeira na forma normal conjuntiva e a segunda na forma normal disjuntiva)

$$(p \vee q \vee \neg r) \wedge (\neg p \vee q \vee r) \qquad \text{e} \qquad p \wedge r \vee \neg p \wedge \neg r \vee q$$

são semanticamente equivalentes mas a segunda fórmula é mais simples que a primeira.

Qualquer juízo semântico onde a primeira fórmula intervenha é igualmente válido se essa fórmula for substituída pela segunda.

Um exemplo de redução semântica é a redução das chamadas **clausulas fechadas**; isto é, clausulas que contêm, simultaneamente, um literal p e a sua negação $\neg p$.

Nas formas conjuntivas uma tal clausula fechada representa uma disjunção do tipo

$$C_f \;=\; (p \vee \neg p \vee \ldots)$$

Semanticamente C_f é sempre equivalente a $\top$.

Nas formas disjuntiva uma clausula fechada representa uma conjunção do tipo

$$C_f \;=\; (p \wedge \neg p \wedge \ldots)$$

Neste caso tem-se $C_f \equiv \bot$.

Uma forma conjuntiva φ que contenha uma clausula fechada C_f tem a forma

$$\varphi \;=\; C_f \wedge \Phi$$

em que Φ denota a conjunção das restantes clausulas. Como $C_f \equiv \top$, qualquer que seja o modelo $\mathcal{M}$, verifica-se

$$\mathcal{M} \models \varphi \quad \text{sse} \quad \mathcal{M} \models \top \quad \& \quad \mathcal{M} \models \Phi$$

Como todo o modelo satisfaz $\top$, conclui-se que $\mathcal{M} \models \varphi$ sse $\mathcal{M} \models \Phi$ o que implica $\varphi \equiv \Phi$.

Dualmente, uma forma disjuntiva φ que contenha uma clausula fechada C_f tem a forma

$$\varphi \;=\; C_f \vee \Phi$$

denotando Φ a disjunção das restantes clausulas. Como, aqui, $C_f \equiv \bot$ usando o mesmo tipo de argumentos concluímos que $\varphi \equiv \Phi$.

Conclui-se, assim,

4 PROPOSIÇÃO (REDUÇÃO DAS CLAUSULAS FECHADAS) *Uma forma normal conjuntiva (ou disjuntiva) é semanticamente equivalente à forma que se obtém retirando-lhe todas as suas clausulas fechadas.*

Por aplicação sucessiva desta redução (ou de outras reduções) pode acontecer que se chegue a uma forma vazia. Neste caso uma **forma conjuntiva completamente fechada** (com todas as clausulas fechada) é semanticamente equivalente a $\top$, enquanto que uma forma disjuntiva vazia é equivalente a $\bot$.

Conclui-se então

5 PROPOSIÇÃO (REDUÇÃO DAS FORMAS FECHADAS) *Uma forma normal conjuntiva fechada é tautologia. Uma forma disjuntiva fechada é inconsistente.*

Uma outra forma de redução semântica é possível quando, em duas clausulas diferentes, ocorrem um símbolo p e a sua negação $\neg p$. Um conjunto de clausulas onde tal ocorra será da forma

$$\{\{p\ ,\ \Phi\}\ ,\ \{\neg p\ ,\ \Phi'\}\ ,\ \ldots\}$$

representando Φ, Φ' os restantes literais dentro de cada uma das clausulas.

O par de clausulas $\{p\ ,\ \Phi\}$ e $\{\neg p\ ,\ \Phi'\}$, que contém p e $\neg p$, diz-se **resolúvel**. O seu **resolvente** é a clausula

$$\{\Phi\ ,\ \Phi'\}$$

que se obtém por união das clausulas resolúveis e retirando os dois literais que provocam a resolução.

Por simplicidade considere-se o caso particular de uma fórmula ϕ, aqui representada na forma normal conjuntiva[3], do tipo

$$\phi\ =\ (p \vee \alpha) \wedge (\neg p \vee \beta)$$

Verifica-se facilmente que, para qualquer modelo $\mathcal{M}$

$$\mathcal{M} \models \phi \quad \text{sse} \quad \begin{cases} \mathcal{M} \models \alpha & , \quad \text{se } p \notin \mathcal{M} \\ \mathcal{M} \models \beta & , \quad \text{se } p \in \mathcal{M} \end{cases} \tag{2}$$

Considere-se a fórmula $\phi \wedge (\alpha \vee \beta)$ que se obtém de ϕ juntando-lhe o resolvente das duas clausulas. É fácil verificar que

$$\mathcal{M} \models \phi \wedge (\alpha \vee \beta) \quad \text{sse} \quad \begin{cases} \mathcal{M} \models \alpha \wedge (\alpha \vee \beta) & , \quad p \notin \mathcal{M} \\ \mathcal{M} \models \beta \wedge (\alpha \vee \beta) & , \quad p \in \mathcal{M} \end{cases} \tag{3}$$

[3] Uma análise perfeitamente dual pode ser feita para formas normais disjuntivas.

e, por conseguinte, $\mathcal{M} \models \phi$ sse $\mathcal{M} \models \phi \wedge (\alpha \vee \beta)$ e pode-se concluir que ϕ e $\phi \wedge (\alpha \vee \beta)$ são semanticamente equivalentes.

Genericamente

6 PROPOSIÇÃO (CLAUSULAS RESOLÚVEIS) *Uma forma normal conjuntiva (ou disjuntiva) é semanticamente equivalente à forma que se obtém juntando-lhe qualquer resolvente de um par de clausulas resolúveis.*

Notas:

1. Aparentemente juntar uma clausula nova a uma forma normal, que contenha um par de clausulas resolúveis $\{p\ ,\ \Phi\}$ e $\{\neg p\ ,\ \Phi'\}$, não contribui para a sua simplificação; o resultado final parece ser mais complexo uma vez que tem mais uma clausula.
   No entanto a nova clausula $\{\Phi\ ,\ \Phi'\}$ elimina p e a sua negação $\neg p$. Pode acontecer que a nova clausula progrida no sentido de se obter a clausula vazia $\{\ \}$; recorde-se que, caso tal clausula apareça, pode-se concluir imediatamente a inconsistência da forma normal.
2. Se este mecanismo for usado para gerar sistematicamente novas clausulas a partir da resolução de clausulas existentes, deve-se usar um método incremental que evite a repetição de clausulas.
   Para isso, no passo n, pelo menos uma das clausulas resolúveis deve ser uma "nova" clausula; isto é, uma clausula que foi criada no passo $n-1$.

A segunda nota levanta a necessidade de se introduzir a noção de **forma fechada por resolução**. Um conjunto de clausulas diz-se **fechado por resolução** quando não contém clausulas fechadas e qualquer eventual resolvente que seja possível gerar já pertence à forma ou então é clausula fechada.

Exemplo: Ambas as formas

$$\{\{a, \neg b\}\ ,\ \{\neg a, b\}\} \qquad \text{e} \qquad \{\{a, b\}\ ,\ \{\neg a, b\}\ ,\ \{b\}\}$$

são fechadas por resolução. A primeira porque qualquer eventual resolvente das duas clausulas (que pode ser feita de duas formas distintas) é uma clausula fechada. A segunda porque a única resolução possível já é uma clausula da forma.

□

Já referimos que uma forma normal conjuntiva que contenha a clausula vazia é necessariamente inconsistente e que uma forma normal disjuntiva contendo a clausula vazia é forçosamente uma tautologia.

O inverso é mais difícil de provar e não iremos procurar fazê-lo aqui. Pode-se enunciar, no entanto, o seguinte resultado:

1 TEOREMA (COMPLETUDE DA RESOLUÇÃO) *Seja ϕ uma forma normal fechada por resolução que não contém a clausula vazia. Então, se for uma forma conjuntiva será verificável (não é inconsistente); se for uma forma disjuntiva será refutável (não é tautologia).*

Este resultado sugere um algoritmo para verificar a inconsistência de formas normais conjuntivas (ou a tautologia de formas normais disjuntivas) baseada numa ideia simples:

Partindo de um conjunto de clausulas $\mathcal{C}$ procede-se do seguinte modo:

1. Simplificar $\mathcal{C}$ removendo todas as clausulas fechadas.
2. Se $\mathcal{C} = \{\ \}$ terminar o algoritmo com a mensagem *Não*.
3. Construir o fecho por resolução de $\mathcal{C}$.
4. Se $\mathcal{C}$ contém a clausula vazia terminar o algoritmo com a mensagem *Sim*; caso contrário, terminar o algoritmo com a mensagem *Não*.

Em termos gerais este é o chamado **algoritmo de resolução de Robinson** aplicado, neste caso, a formas proposicionais.

exemplo: Considere-se a forma

$$\{\{\neg a\}\ ,\ \{\, a\ ,\ \neg b\}, \{b\}\}$$

É possível juntar a esta forma dois novos resolventes

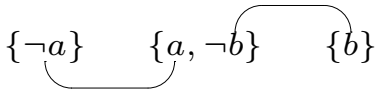
$$\{\{\neg b\}\ ,\ \{a\}\}$$

Resolvendo cada uma destas novas clausulas com as clausula iniciais obtém-se dois novos resolventes

$$\{\{\ \}\ ,\ \{\ \}\}$$

A presença da clausula vazia assegura que a forma inicial é inconsistente.

A construção de resolventes não necessita de uma representação explícita. É possível representar os resolventes por "ligações" (como se indica na figura seguinte).



A figura indica que, no conjunto de clausulas afectadas pelas ligações, todos os literais ocorrem numa "ligação aberta"; isso indica que os todos os literais são removíveis e, portanto, a clausula vazia acaba por ser construída.

Se a forma inicial fosse

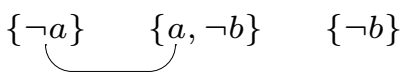
$$\{\{\neg a\}\ ,\ \{\ a\ ,\ \neg b\},\{\neg b\}\}$$

existiria um novo resolvente

$$\{\{\neg b\}\}$$

e, não sendo possível construir outros resolventes e não se tendo construído a clausula vazia, temos de concluir que a forma inicial é verificável.

Neste exemplo as ligações seriam

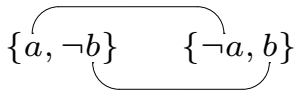


Não sendo possível ligar todos os literais, do resolvente não resulta a clausula vazia.

É importante notar que nem sempre ter todos os literais ligados implica um resolvente vazio; por exemplo na forma

$$\{\{a\ ,\ \neg b\}\ ,\ \{\neg a\ ,\ b\}\}$$

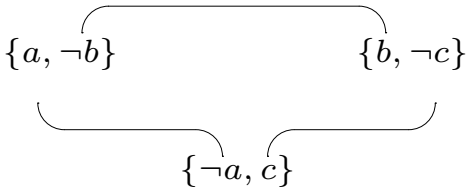
resultariam as ligações a seguir indicadas



As duas ligações representam os dois resolventes possíveis: $\{a, \neg a\}$ e $\{b, \neg b\}$. Ambos são clausulas fechadas que não afectam a validade da forma; não contribuem para o progresso do algoritmo de resolução. Este facto é detectado pelo facto das ligações formarem um ciclo.

Outro exemplo de ciclo, mais complexo, é o que resulta da forma

$$\{\{a, \neg b\} \ , \ \{b, \neg c\} \ , \ \{c, \neg a\}\}$$



O ciclo é indicativo de resolventes que são clausulas fechadas.

O facto de ser $\phi \equiv \varphi$ permite substituir uma fórmula pela outra em qualquer juízo onde ocorra. Assim, por exemplo, ϕ será inconsistente (ou tautologia, ou verificável, etc) se e só se φ for inconsistente (ou tautologia, ou verificável, etc).

Muitas vezes porém não se é tão exigente e pretende-se apenas substituir uma fórmula por outra no contexto de um juízo particular. Por exemplo, é frequente pretender-mos verificar se uma forma ϕ é inconsistente substituindo-a por uma outra forma ϕ' e procurando ver se essa é uma forma inconsistente.

O algoritmo de resolução de Robinson é bastante ineficiente quando se procura apenas verificar a inconsistência de formas proposicionais. Um algoritmo melhor é baseado no chamado **"split" de Davis-Putnam**.

Seja Φ uma proposição que contém o símbolo p. Um **fraccionamento** ("split") por p é um par de formas $\langle \Phi_p^+ \ , \ \Phi_p^- \rangle$ tais que:

(i) Nenhuma das formas Φ_p^+ ou Φ_p^- contém p, e

(ii) $\Phi \ \equiv \ (p \rightarrow \Phi_p^+ ; \Phi_p^-)$

Conclui-se facilmente

2 TEOREMA (DAVIS-PUTNAM) *Nas condições anteriores Φ é inconsistente (respectivamente, tautologia) se e só se ambas as formas Φ_p^+ e Φ_p^- forem inconsistente (respectivamente, tautologias).*

Prova:

Note-se que, para todo o modelo $\mathcal{M}$ se verifica

$$\mathcal{M} \models \Phi \text{ sse } \begin{cases} \mathcal{M} \setminus \{p\} \models \Phi_p^+ & , \quad \text{se } p \in \mathcal{M} \\ \mathcal{M} \setminus \{p\} \models \Phi_p^- & , \quad \text{se } p \notin \mathcal{M} \end{cases} \tag{4}$$

Vamos supor, primeiro, que Φ_p^+ e Φ_p^- são ambas formas inconsistente.

Tomando um qualquer modelo $\mathcal{M}$ e tentando calcular $\mathcal{M} \models \Phi$ usando (4) verifica-mos que se tinha de verificar $\mathcal{M} \setminus \{p\} \models \Phi_p^+$ ou $\mathcal{M} \setminus \{p\} \models \Phi_p^-$ (consoante o modelo contivesse ou não p); isso não é possível porque, por hipótese, ambas estas formas são inconsistente e, assim, não existe nenhum modelo que as valide.

Inversamente, vamos supor que Φ que é inconsistente e que se pretende, num modelo particular $\mathcal{M}$, determinar $\mathcal{M} \models \Phi_p^+$ ou $\mathcal{M} \models \Phi_p^-$; note-se que, porque Φ_p^+ e Φ_p^- não contêm p, só são relevantes os modelos sem p.

Só seria possível ocorrer $\mathcal{M} \models \Phi_p^+$ se, no modelo $\mathcal{M} \cup \{p\}$, a forma Φ fosse verificada; o que nunca acontece porque é inconsistente. O mesmo se passa se procurássemos verificar $\mathcal{M} \models \Phi_p^-$; aqui teria de se verificar $\mathcal{M} \models \Phi$ que, mais uma vez, não é possível porque Φ é inconsistente.

O mesmo tipo de argumentos se aplica se o juízo em causa for *"é tautologia"*.

exemplo: Considere-se a fórmula $\Phi = (a \vee b) \wedge c$ escrita como uma forma conjuntiva.

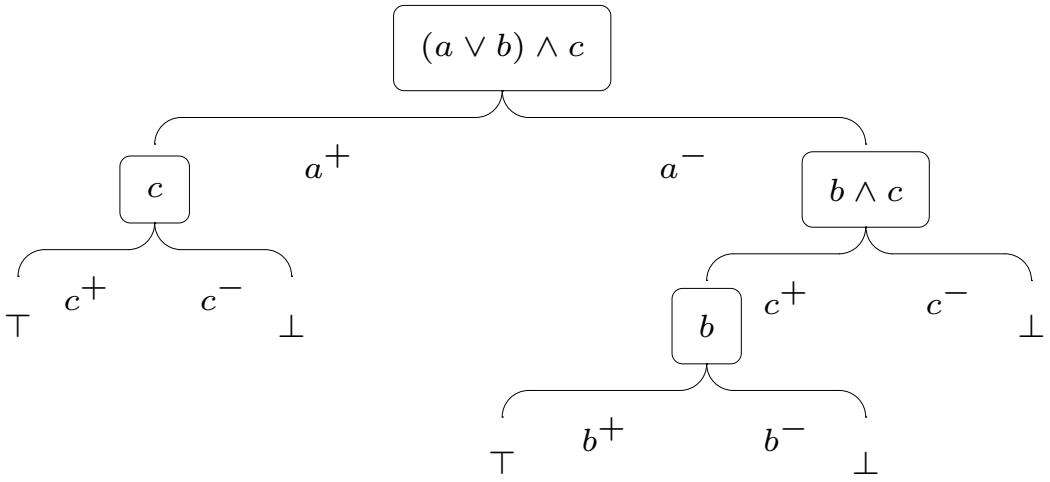
Fazendo o fraccionamento pelo símbolo a construímos

$$\Phi_a^+ = c \qquad \Phi_a^- = b \wedge c$$

simplificando a fórmula original na suposição que a é válido (para o primeiro caso) e na suposição que a não é válido (no segundo caso).

As novas fórmulas não contêm o símbolo de fraccionamento a. Agora, a cada uma delas e do mesmo modo, pode-se aplicar a operação de "split" com um dos símbolos restantes.

A seguinte árvore resume todas estas operações de "split", primeiro com a, depois com c e, finalmente, com b. O ramo marcado com a^+ aponta para Φ_a^+ e, reciprocamente, o ramo marcado com a^- aponta para Φ_a^-. De forma análoga estrutura-se todos os restantes "splits".



Na árvore os caminhos que terminam em folhas $\top$ determinam os modelos que validam a fórmula; os caminhos que terminam em folhas $\bot$ definem os modelos onde a fórmula não é válida. A fórmula só seria inconsistente se todos os caminhos terminassem em $\bot$.

A análise desta árvore permite reconhecer, como modelos que validam a fórmula,

$$\{a = 1 \ , \ c = 1 \ , \ b = ?\} \cup \{a = 0 \ , \ c = 1 \ , \ b = 1\}$$

que correspondem aos caminhos terminados em $\top$. Estes modelos podem ser representados por uma forma normal disjuntiva

$$\Phi' \;=\; (a \wedge c) \vee (\neg a \wedge c \wedge b)$$

que é, portanto, semanticamente equivalente à forma conjuntiva inicial.

---

7 PROPOSIÇÃO ("SPLIT" DE CNF) *Dada uma forma normal conjuntiva Φ contendo o símbolo p,*

(i) Seja Φ_p^+ a forma que se obtém eliminando todas as cláusulas que contêm p e eliminando $\neg p$ de todas as clausulas que o contêm; então

$$p \wedge \Phi_p^+ \;\equiv\; p \wedge \Phi$$

(ii) Seja Φ_p^- a forma que se obtém eliminando todas as cláusulas que contêm $\neg p$ e eliminando p de todas as clausulas que o contêm; então

$$\neg p \wedge \Phi_p^- \;\equiv\; \neg p \wedge \Phi$$

(iii) $\langle \Phi_p^+ \;,\; \Phi_p^- \rangle$ forma um par "split" de Φ por p; isto é, não contêm p e verifica-se $\Phi \;\equiv\; p \longrightarrow \Phi_p^+ ; \Phi_p^-$.

Considere-se uma clausula $(p \vee \alpha)$ de Φ contendo p; α é qualquer e Φ' é a formula que se obtém eliminado essa clausula; isto é, $\Phi = (p \vee \alpha) \wedge \Phi'$. Então

$$p \wedge \Phi \;=\; p \wedge (p \vee \alpha) \wedge \Phi' \;\equiv\; p \wedge \Phi'$$

Considere-se uma clausula $(\neg p \vee \beta)$ de Φ contendo $\neg p$; tem -se $\Phi = (\neg p \vee \beta) \wedge \Phi'$ para algum Φ'. Então

$$p \wedge \Phi \;=\; p \wedge (\neg p \vee \beta) \wedge \Phi' \;\equiv\; p \wedge \beta \wedge \Phi'$$

Estas duas equivalências provam *(i)*; a observação *(ii)* prova-se de forma dual. A condição *(iii)* é uma consequência das duas primeiras

$$\Phi \equiv p \wedge \Phi \vee \neg p \wedge \Phi \equiv p \wedge \Phi_p^+ \vee \neg p \wedge \Phi_p^- = p \longrightarrow \Phi_p^+ ; \Phi_p^-$$

e da observação que nenhuma das formas Φ_p^+ ou Φ_p^- contém p.

1 COROLÁRIO *Se p forma uma* **clausula unitária** *(i.e. uma clausula que só contém esse literal) então $\Phi_p^- \equiv \bot$ e Φ é inconsistente se e só se se Φ_p^+ for inconsistente. Se for $\neg p$ uma clausula unitária então $\Phi_p^+ \equiv \bot$ e Φ é inconsistente se e só se Φ_p^- for inconsistente.*
Se p for **literal puro** *(i.e. se $\neg p$ não ocorrer em nenhuma outra clausula) então Φ é inconsistente se e só se Φ_p^+ for inconsistente. Se $\neg p$ for literal puro (p não ocorre em nenhuma clausula) então Φ é inconsistente se e só se Φ_p^- for inconsistente.*

No calculo de Φ_p^-, sendo $\{p\}$ é uma clausula unitária, ao retira-se-lhe p ela fica vazia; com uma clausula vazia, uma forma conjuntiva é equivalente a $\bot$; neste caso tem-se $\Phi \equiv p \wedge \Phi_p^+$. De forma dual, no calculo de Φ_p^+, se existir a clausula unitária $\{\neg p\}$, ela ficará vazia depois de lhe ser retirado esse literal; neste caso $\Phi_p^+ \equiv \bot$ e, portanto, $\Phi \equiv \neg p \wedge \Phi_p^-$.

Se $\neg p$ não existir em nenhuma clausula (p é um literal puro), para o calculo de Φ_p^+ apenas se retiram todas as clausulas que contêm p. Portanto pode-se escrever $\Phi = \mathcal{C} \wedge \Phi_p^+$, sendo $\mathcal{C}$ o conjunto das clausulas que contêm p; para efeitos de teste da inconsistência de Φ basta testar se Φ_p^+ é inconsistente.

Da mesma forma, se p não existir em Φ, então pode-se escrever $\Phi = \mathcal{C} \wedge \Phi_p^-$, sendo $\mathcal{C}$ as clausulas que contêm $\neg p$; neste caso, Φ é inconsistente se e só se Φ_p^- for inconsistente.

exemplo: Considere-se

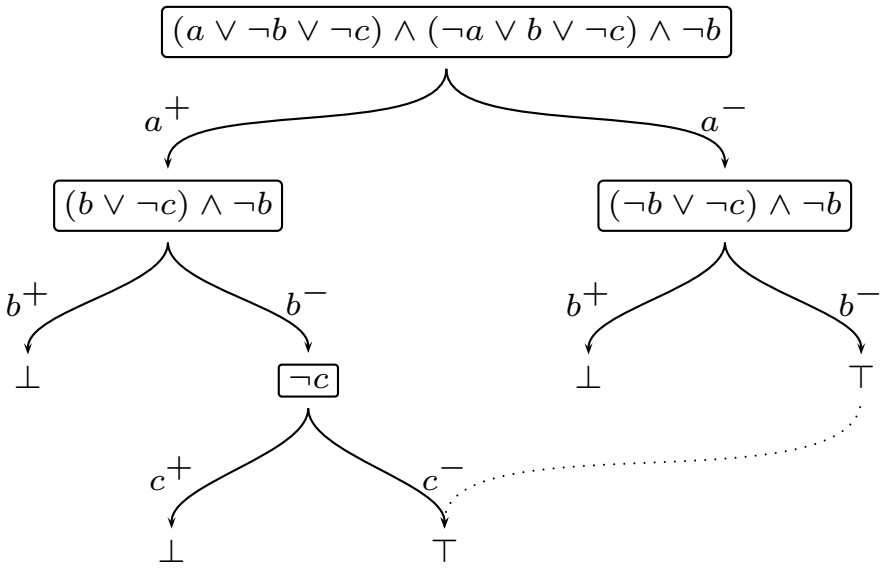
$$\Phi = (a \vee \neg b \vee \neg c) \wedge (\neg a \vee b \vee \neg c) \wedge \neg b$$

Calcula-se Φ_a^+ começando por eliminar a 1ª clausula (porque contém a) e eliminar $\neg a$ da 2ª clausula. Para calcular Φ_a^- elimina-se a 2ª clausula (porque contém $\neg a$) e elimina-se a da 1ª clausula. Logo

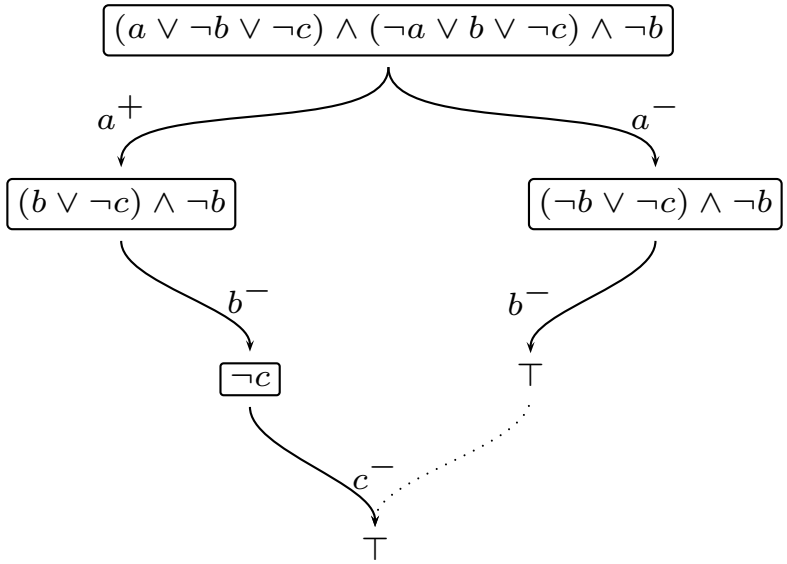
$$\Phi_a^+ = (b \vee \neg c) \wedge \neg b \qquad \Phi_a^- = (\neg b \vee \neg c) \wedge \neg b$$

Ambas as formas contêm a clausula unitária $\{\neg b\}$. Fraccionando por b,

$$(\Phi_a^+)_b^+ = \bot \quad (\Phi_a^+)_b^- = \neg c \quad (\Phi_a^-)_b^+ = \bot \quad (\Phi_a^-)_b^- = \top$$



De facto trata-se de um grafo acíclico se "juntar-mos" todos os ramos que ligam ao mesmo nó (exceptuando, desta regra, os nodos "inconsistência" $\bot$). Se tornar-mos implícitos os nodos $\bot$ o grafo ainda pode ser mais simplificado

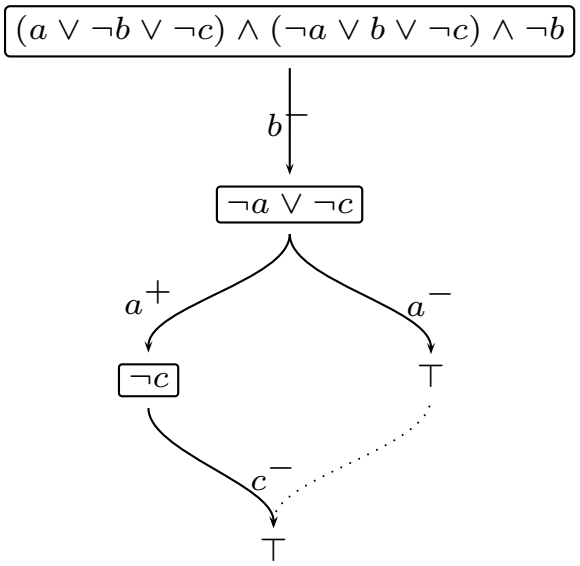


Este grafo torna evidentes os modelos que validam a fórmula e que, como se sabe, são determinados pelos caminhos que terminam em $\top$. Neste caso serão

$$\{a = 1 \ , \ b = 0 \ , \ c = 0\} \cup \{a = 0 \ , \ b = 0 \ , \ c = ?\}$$

O grafo ilustra também que o símbolo a foi a pior escolha possível para iniciar o fraccionamento. De facto, examinando a forma inicial, verifica-se que $\neg b$ determina uma clausula unitária e que c é um literal puro. Qualquer destes dois símbolos teria sido uma melhor escolha.

Iniciando o fraccionamento por b, usando o corolário da proposição 7, poder-se-ia ter directamente



O fraccionamento iniciado com c é ainda mais simples; usando o facto de ser um literal puro, a inconsistência da fórmula inicial resume-se ao teste da inconsistência de $\Phi_c^- = \neg b$.

1.5. Modelos e Diagramas de Decisão Binária

Subentende-se, no que se segue um conjunto finito de símbolos proposicionais $\mathcal{P}$ e a linguagem $\mathcal{L}_{\mathcal{P}}$ por eles gerada. Representamos por $\mathcal{M}_{\mathcal{P}} = \wp(\mathcal{P})$ o conjunto de todos os subconjuntos de $\mathcal{P}$, i.e. os modelos de $\mathcal{L}_{\mathcal{P}}$.

A **relação de validade** $\omega \models \phi$ estabelece quando um modelo $\omega \in \mathcal{M}_{\mathcal{P}}$ valida uma fórmula $\phi \in \mathcal{L}_{\mathcal{P}}$. Como elemento de $\mathcal{L}_{\mathcal{P}} \times \mathcal{M}_{\mathcal{P}}$ pode ser representada de várias formas; nomeadamente como a função $\Omega : \mathcal{L}_{\mathcal{P}} \to \wp(\mathcal{M}_{\mathcal{P}})$, que associa cada *fórmula* ao *conjunto de modelos* que a validam,

$$\Omega(\phi) \doteq \{ \omega \mid \omega \models \phi \} \tag{5}$$

ou a função $\mho : \mathcal{M}_{\mathcal{P}} \to \wp(\mathcal{L}_{\mathcal{P}})$, que associa um *modelo* ω ao *conjunto de fórmulas* que valida

$$\mho(\omega) \doteq \{ \phi \mid \omega \models \phi \} \tag{6}$$

Em termos algébricos a função Ω é a **representação canónica** da lógica $\mathcal{L}_{\mathcal{P}}$ vista como uma álgebra booleana. Para ver o que isto significa, note-se[4] as seguintes propriedades de Ω

(i) $\Omega(\phi \wedge \varphi) = \Omega(\phi) \cap \Omega(\varphi)$ e $\Omega(\top) = \mathcal{M}_{\mathcal{P}}$

(ii) $\Omega(\phi \vee \varphi) = \Omega(\phi) \cup \Omega(\varphi)$ e $\Omega(\bot) = \emptyset$

(iii) $\Omega(\neg\phi) = \mathcal{M}_{\mathcal{P}} \setminus \Omega(\phi)$

3 TEOREMA (REPRESENTAÇÃO CANÓNICA DE $\mathcal{L}_{\mathcal{P}}$) *Se $\mathcal{P}$ for um conjunto finito, a função $\Omega : \mathcal{L}_{\mathcal{P}} \to \wp(\mathcal{M}_{\mathcal{P}})$ é um isomorfismo, a menos da equivalência semântica, de álgebras booleanas.*

[4] A demonstração é uma consequência simples da definição da relação de validade.

A prova deste resultado é uma consequência dos dois lemas seguintes. O primeiro estabelece que Ω é um homomorfismo de álgebras booleanas (i.e. um morfismo que preserva a estrutura algébrica) e que é uma função injectiva. O segundo lema estabelece que esta função também é também sobrejectiva.

Em ambos os resultados as fórmulas são vistas "a menos da equivalência semântica"; isto significa que duas fórmulas semanticamente equivalente são consideradas como sendo "a mesma" fórmula.

1 FACTO $\Omega(\cdot)$ *é uma* **imersão** *de álgebras booleanas.*

2 FACTO *Dado* $\zeta \subseteq \wp(\mathcal{P})$ *existe* ϕ *tal que* $\Omega(\phi) = \zeta$.

Vendo $\wp(\mathcal{M}_{\mathcal{P}})$, como uma álgebra booleana de conjuntos (com as operações de soma $\cup$, produto $\cap$, complemento $\setminus\cdot$, topo $\mathcal{M}_{\mathcal{P}}$ e zero $\emptyset$) estas propriedades confirmam que a função Ω é um homomorfismo de álgebras booleanas.

Adicionalmente Ω é uma função injectiva já que a definição de equivalência semântica pode ser escrita

$$\phi \equiv \varphi \qquad \text{sse} \qquad \Omega(\phi) = \Omega(\varphi) \tag{7}$$

Considere-se, agora, um qualquer modelo $\omega \in \mathcal{M}_{\mathcal{P}}$ e a fórmula ϕ_ω definida por

$$\phi_\omega \;=\; \bigwedge_{p \in \omega} p \;\wedge\; \bigwedge_{q \notin \omega} \neg q \tag{8}$$

Então $\upsilon \models \phi_\omega$ se e só se, para todo $p \in \omega$, $\upsilon \models p$ (i.e. $p \in \upsilon$) e, para todo $q \notin \omega$, $\upsilon \models \neg q$ (i.e. $q \notin \upsilon$). Portanto $\upsilon \models \phi_\omega$ se e só se $\upsilon = \omega$.

$$\Omega(\phi_\omega) \;=\; \{\, \upsilon \mid \upsilon \models \phi_\omega \,\} \;=\; \{\omega\}$$

A fórmula ϕ_ω pode ser vista como uma clausula de uma forma normal disjuntiva que tem ω como único modelo.

Considere-se agora um conjunto de modelos $\zeta \in \wp(\mathcal{M}_{\mathcal{P}})$; procuramos construir um ϕ tal que $\Omega(\phi) = \zeta$. Defina-se $\phi \doteq \bigvee_{\omega \in \zeta} \phi_\omega$ como uma forma normal disjuntiva formada gerada pelas clausulas ϕ_ω. Então

$$\Omega(\phi) = \Omega(\bigvee_{\omega \in \zeta} \phi_\omega) = \bigcup_{\omega \in \zeta} \Omega(\phi_\omega) \;=\; \bigcup_{\omega \in \zeta} \{\omega\} \;=\; \zeta$$

exemplo: Seja $\mathcal{P} = \{a, b, c, d\}$ e tomemos um modelo qualquer de $\mathcal{L}_{\mathcal{P}}$. Por exemplo

$$\omega = \{a, c\}$$

A fórmula (única, a menos de equivalência semântica) que tem por representação o conjunto singular $\{\omega\}$ será

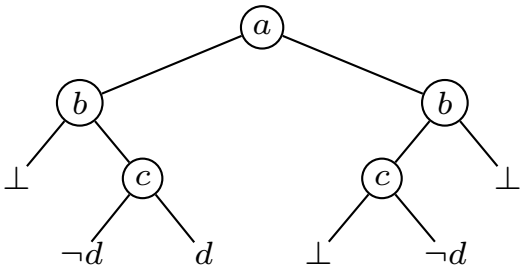
$$\phi_\omega \quad = \quad a \wedge \neg b \wedge c \wedge \neg d$$

Nenhum outro modelo pode validar esta fórmula; isto porque um modelo diferente teria de conter c ou d ou, então, não conter a ou não conter b.

Tomemos o conjunto de modelos $\{\{a, c\}, \{b\}, \{a, d\}\}$.
A fórmula que tem esta representação será

$(a \wedge \neg b \wedge c \wedge \neg d) \vee (\neg a \wedge b \wedge \neg c \wedge \neg d) \vee (a \wedge \neg b \wedge \neg c \wedge d)$

Claramente que esta não é a mais simples representação desta fórmula e seria possível simplificá-la numa árvore.



O ponto seguinte é a apresentação destas representações $\zeta \in \wp(\mathcal{M}_{\mathcal{P}})$. Tomemos um exemplo

$$\zeta = \{\{a, b, c\}, \{b, d\}, \{a, c\}, \{c, d\}\}$$

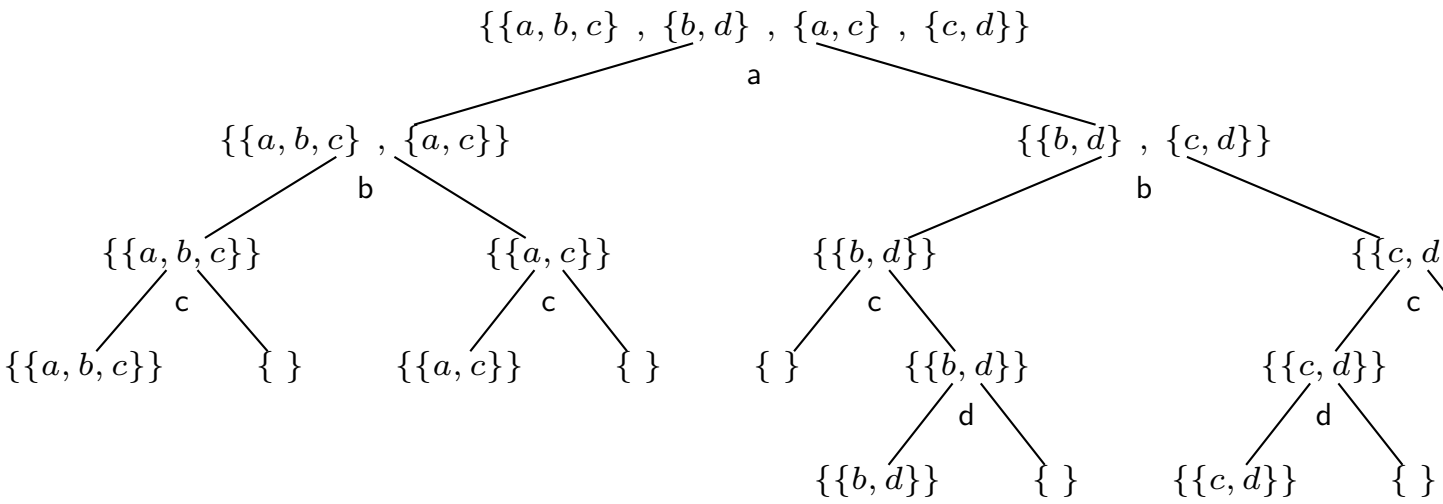
Pode-se dividir este conjunto em dois: o conjunto dos modelos que contêm a e o conjunto dos que não contêm esse símbolo

$$\{\{a, b, c\}, \{a, c\}\} \cup \{\{b, d\}, \{c, d\}\}$$

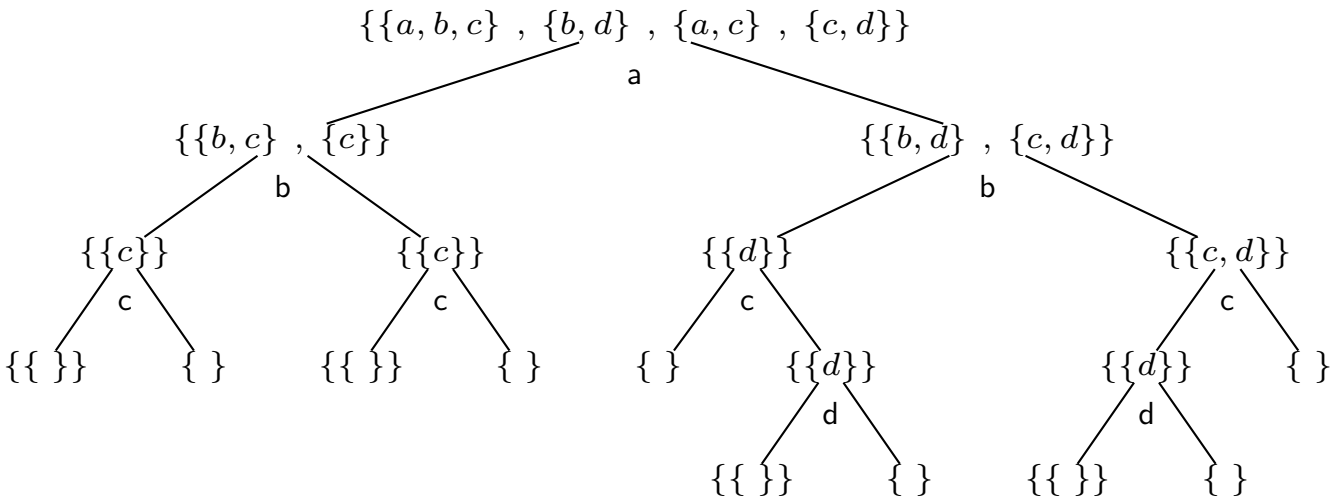
Cada uma destas componentes pode ser dividida em duas: a parte que contêm b e a parte que não contém esse símbolo.

$$\left(\{\{a, b, c\}\} \cup \{\{a, c\}\}\right) \cup \left(\{\{b, d\}\} \cup \{\{c, d\}\}\right)$$

E assim sucessivamente; para controlar a complexidade da apresentação é conveniente apresentar esta estrutura através de uma árvore



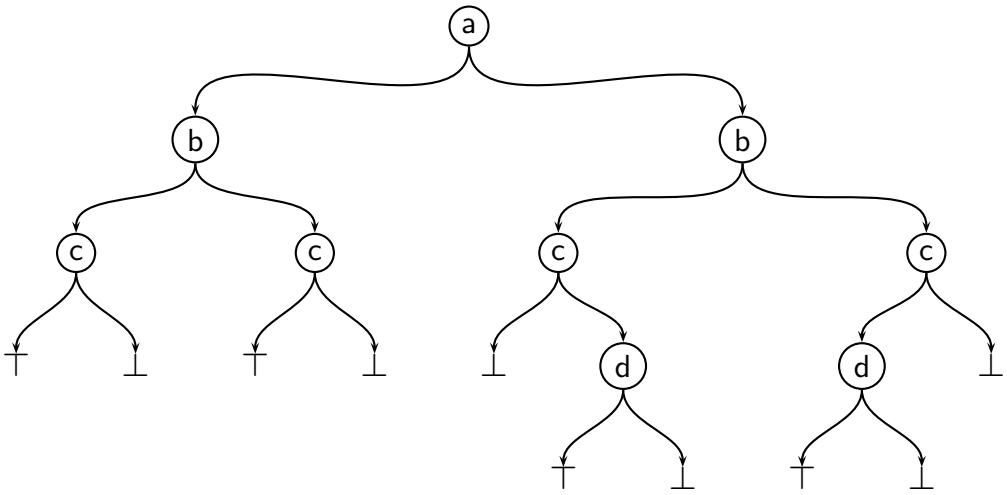
Simplifica-se a árvore não apresentando explicitamente nos "conjuntos esquerdos" o símbolo que determina o fraccionamento.



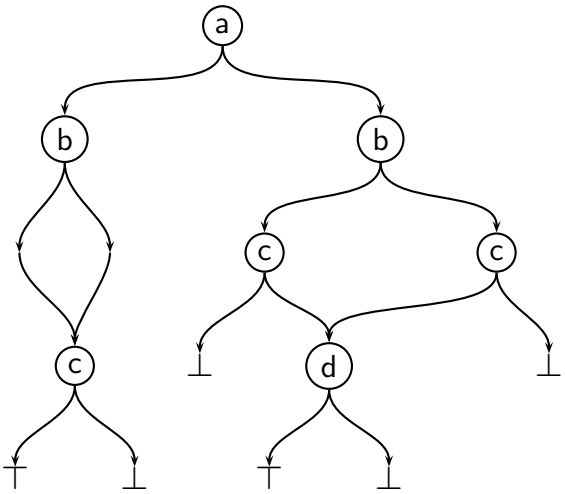
Pode-se ainda simplificar esta construção segundo os seguintes princípios:

(i) As folhas são uma de duas representações: a **representação vazia** $\{\ \}$ (escrita como $\bot$) ou a **representação singular** $\{\{\ \}\}$ (escrita como $\top$).

(ii) Os nodos da árvore são marcados apenas com o símbolo proposicional; a subárvore esquerda apresenta a componente da representação onde esse símbolo aparece; a sub-árvore direita apresenta a componente onde o símbolo não aparece.



Finalmente simplifica-se ainda esta apresentação transformando a árvore num grafo acíclico onde são partilhadas as "sub-árvores repetidas"



Esta apresentação assume a forma de um **Diagrama de Decisão Binária** (ou **DDB**) que pode ser definido do seguinte modo.

4 DEFINIÇÃO *Seja $\mathcal{S}$ um conjunto finito de símbolos nos quais está definida uma ordem total. Um* **diagrama de decisão binária tipado** *é um par $\langle \mu , \mathcal{G} \rangle$ formado por um* **sinal** *$\mu \in \{+, -\}$ e um grafo $\mathcal{G} = \langle \mathcal{N}, \mathcal{E} \rangle$, acíclico e construído segundo uma das seguintes possibilidades:*

(i) *$\mathcal{G}$ é formado por um só nodo marcado com um dos símbolos especiais $\top$ ou $\bot$.*

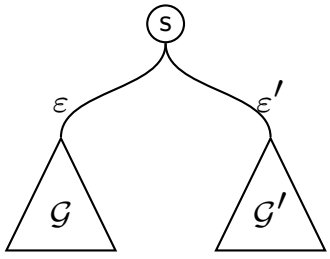
(ii) *$\mathcal{G}$ é um triplo escrito como $\langle s \longrightarrow \Phi \,;\, \Psi \rangle$, com $s \in \mathcal{S}$, Φ e Ψ são DDB que contêm apenas símbolos s' menores do que s.*
O grafo $\mathcal{G}$ designa-se, neste caso, por **vértice de raiz s** *e é formado por um nodo marcado com s (a raiz), pelos vértices associados a Φ e a Ψ e por dois ramos com origem s e destino nos vértices de Φ e Ψ marcados com os respectivos sinais.*

No grafo da página 32 considerou-se a ordem $a > b > c > d$ e, para facilitar a leitura, repetiu-se os nodos $\bot$ e $\top$.

Se Φ for um DDB designa-se por $\bar{\Phi}$ o DDB que se obtém trocando o sinal de Φ.

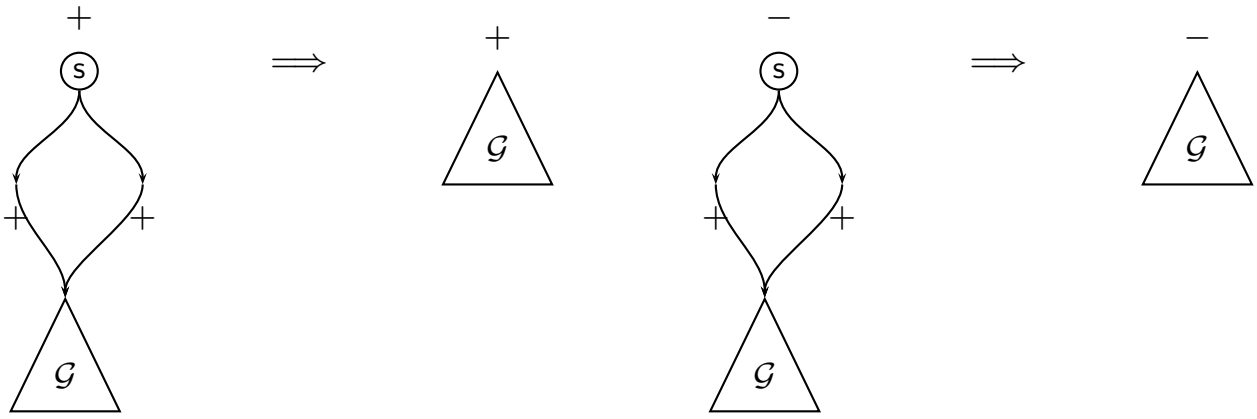
5 DEFINIÇÃO *Um vértice $\mathcal{G}$ está* **reduzido** *se não contém sub-vértices repetidos nem nenhum sub-vértice $\mathcal{G}'$ da forma $\langle s \longrightarrow \Phi; \Phi \rangle$. Um vértice está* **normalizado** *se está reduzido, os sub-DDB estão normalizados, nenhum nodo é marcado $\bot$ e o sub-DDB esquerdo tem sinal $+$.*

O diagrama seguinte esquematiza um vértice com raiz s, sub-vértices $\mathcal{G}$, $\mathcal{G}'$ e ramos marcados com os sinais $\varepsilon, \varepsilon'$.



Redução de DDB: Ocorre em vértices da forma $\mathcal{G} = \langle s \rightarrow \Phi\,;\,\Phi\rangle$. O DDB $\langle\,\mu\,,\,\mathcal{G}\,\rangle$ reduz-se a Φ, se for $\mu = +$, ou a $\bar{\Phi}$ se for $\mu = -$. Em qualquer dos casos "faz desaparecer" o símbolo s.

$$+\langle s \rightarrow \Phi\,;\,\Phi\rangle \quad \rightsquigarrow \quad \Phi \qquad , \qquad -\langle s \rightarrow \Phi\,;\,\Phi\rangle \quad \rightsquigarrow \quad \bar{\Phi}$$



Normalização de um DDB

Não Normalizado	Normalizado
$+\bot$	$-\top$
$-\bot$	$+\top$
$+\langle s \rightarrow -\phi\,;\,-\varphi\rangle$	$-\langle s \rightarrow +\phi\,,\,+\varphi\rangle$
$+\langle s \rightarrow -\phi\,;\,+\varphi\rangle$	$-\langle s \rightarrow +\phi\,;\,-\varphi\rangle$
$-\langle s \rightarrow -\phi\,;\,-\varphi\rangle$	$+\langle s \rightarrow +\phi\,;\,+\varphi\rangle$
$-\langle s \rightarrow -\phi\,;\,+\varphi\rangle$	$+\langle s \rightarrow +\phi\,;\,-\varphi\rangle$

DDB servem para apresentar fórmulas e representações de fórmulas. Para isso usam-se as seguintes regras:

DDB como apresentação de uma fórmula

1. Os vértices especiais $\top$ e $\bot$ definem as fórmulas representadas por esses símbolos.

2. O vértice $\mathcal{G} = \langle s \rightarrow \Phi \,;\, \Psi \rangle$ define a fórmula

$$\mathcal{F}_{\mathcal{G}} \;=\; (s \wedge \mathcal{F}_{\Phi} \vee \neg s \wedge \mathcal{F}_{\Psi})$$

   sendo $\mathcal{F}_{\Phi}$ e $\mathcal{F}_{\Psi}$ as fórmulas representadas pelos DDB Φ e Ψ.
3. O DDB $\langle\, \mu \,,\, \mathcal{G} \,\rangle$ define a fórmula $\mathcal{F}_{\mathcal{G}}$ se for $\mu = +$ ou a fórmula $\neg\mathcal{F}_{\mathcal{G}}$ se for $\mu = -$

DDB como apresentação de representações

Os conjuntos de modelos representados por um DDB têm, por referência, um determinado universo $\mathcal{U} \subseteq \mathcal{M}_{\mathcal{P}}$. Vamos representar por $\mathcal{U}_s \subseteq \mathcal{U}$ o sub-conjunto formado pelos modelos que só contêm símbolos $s' < s$.

Em referência ao universo de modelos $\mathcal{U}$

1. O vértice $\top$ determina a representação $\mathcal{U}$; o vértice $\bot$ determina a representação vazia $\{\ \}$.
2. O vértice $\mathcal{G} = \langle s \rightarrow \Phi \,;\, \Psi \rangle$ define a representação $\mathcal{R}_{\mathcal{G}}$

$$\mathcal{R}_{\mathcal{G}} \doteq \mathcal{R}_{\Psi} \,\cup\, \{\, \omega \cup \{s\} \;\mid\; \omega \in \mathcal{R}_{\Phi} \,\}$$

   em que $\mathcal{R}_{\Phi}$ e $\mathcal{R}_{\Psi}$ são as representações definidas pelos DDB Φ e Ψ tendo por referência o universo $\mathcal{U}_s$
3. Seja $\mathcal{R}_{\mathcal{G}}$ o conjunto de modelos que o vértice $\mathcal{G}$ representa. $+\mathcal{G}$ determina a representação $\mathcal{U}/\mathcal{R}_{\mathcal{G}}$ [5]. $-\mathcal{G}$ define a representação $\mathcal{U} \setminus (\mathcal{U}/\mathcal{R}_{\mathcal{G}})$.

A construção de DDB pode ser realizada incrementalmente a partir de outros DDB.

Se $X, Y \in \wp(\mathcal{M}_{\mathcal{P}})$ são descritos pelos DDB Φ_X e Φ_Y então

[5] Se $\mathcal{U}$ é um universo onde está definida uma ordem parcial $\leq$ e $X \subseteq \mathcal{U}$, define-se $\mathcal{U}/X \doteq \{\, y \in \mathcal{U} \mid x \leq y \text{ para algum } x \in X \,\}$.

1. *Complemento:* $\mathcal{M}_{\mathcal{P}} \setminus X$ é descrito por $\bar{\Phi}_X$.
2. *União e Intersecção:* representação de $X \circ Y$, com $\circ \in \{\cap\ ,\ \cup\}$,

(i) Se $\Phi_X = +(a \to \Phi'_X; \Phi''_X)$ e $\Phi_Y = +(a \to \Phi'_Y; \Phi''_Y)$

$$\Phi_{X \circ Y} = (a \to \Phi'_X \circ \Phi'_Y; \Phi''_X \circ \Phi''_Y)$$

(ii) Se $\Phi_X = -(a \to \Phi'_X; \Phi''_X)$ e $\Phi_Y = -(a \to \Phi'_Y; \Phi''_Y)$

$$\Phi_{X \circ Y} = (a \to \bar{\Phi}'_X \circ \bar{\Phi}'_Y; \bar{\Phi}''_X \circ \bar{\Phi}''_Y)$$

(iii) Se $\Phi_X = +(a \to \Phi'_X; \Phi''_X)$ e $\Phi_Y = -(a \to \Phi'_Y; \Phi''_Y)$

$$\Phi_{X \circ Y} = (a \to \Phi'_X \circ \bar{\Phi}'_Y; \Phi''_X \circ \bar{\Phi}''_Y)$$

(iv) Se $\Phi_X = +(a \to \Phi'_X; \Phi''_X)$ e $\text{raiz}(\Phi_Y) < a$.

$$\Phi_{X \circ Y} = (a \to \Phi'_X \circ \Phi_Y; \Phi''_X \circ \Phi_Y)$$

(v) Se $\Phi_X = -(a \to \Phi'_X; \Phi''_X)$ e $\text{raiz}(\Phi_Y) < a$

$$\Phi_{X \circ Y} = (a \to \bar{\Phi}'_X \circ \Phi_Y; \bar{\Phi}''_X \circ \Phi_Y)$$

(vi) *Casos de paragem*

$$\Phi \cup +\top = +\top \qquad \Phi \cup -\top = \Phi$$
$$\Phi \cap -\top = -\top \qquad \Phi \cap +\top = \Phi$$

Subentende-se que, após esta construção, o DDB resultante pode não estar reduzido e/ou normalizado. Por isso cada uma destas construções deve ser seguida por redução e normalização do resultado.

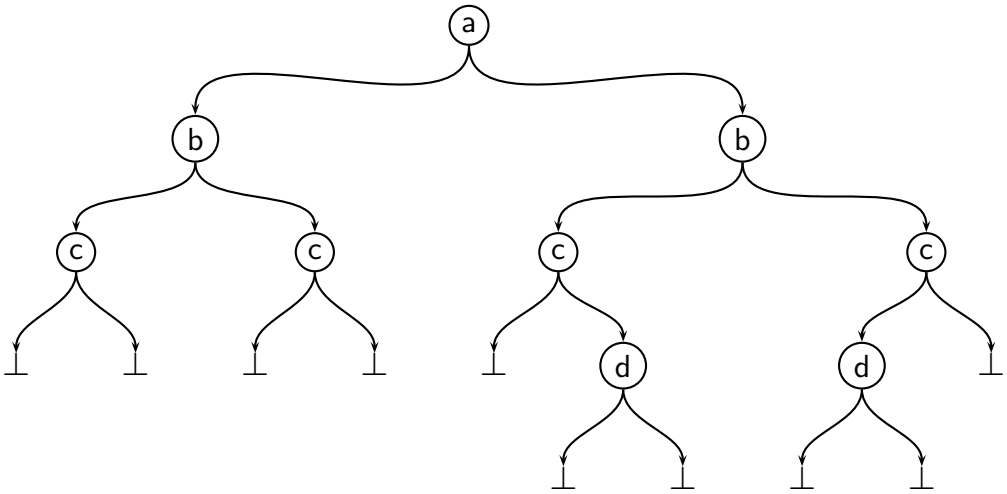
A importância desse passo de simplificação deriva do seguinte resultado.

8 PROPOSIÇÃO *Um DDB representa o conjunto vazio de modelos* $\{\}$ *se e só se pode ser reduzido ao DDB* $-\top$.

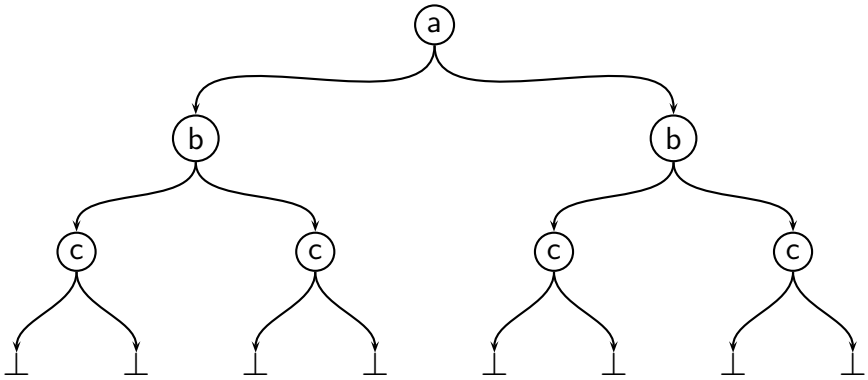
Nota:

Pela construção anterior, é óbvio que o DDB $-\top$ representa o conjunto vazio de modelos. Já não é tão óbvia a implicação inversa.

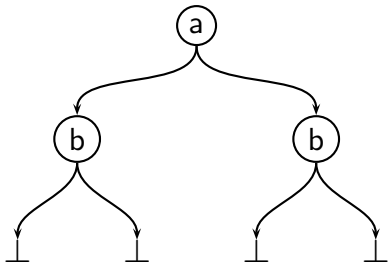
Tomemos, por simplicidade, um DDB não tipado (não existem sinais e o símbolo $\bot$ ocorre). Se for equivalente a $\bot$, todos os caminhos terminam em $\bot$. É o caso do seguinte exemplo.



Começando pelas sub-árvores cujas descendentes são só folhas, e atendendo que todas as folhas são (por hipótese) $\bot$, será possível reduzir essas sub-árvores. Neste exemplo começa-se por reduzir as árvores de raiz d.



Seguem-se as quatro árvores de raiz c.



E assim sucessivamente acabamos por reduzir a árvore a $\bot$.

O último passo consiste simultânea da representação $\Omega(\phi)$ de uma fórmula proposicional arbitrária ϕ e do DDB que descreve essa representação. O processo é indutivo na estrutura das fórmulas:

1. O literal a é representado por $+(a \rightarrow +\top; -\top)$. O literal $\neg a$ é representado por $-(a \rightarrow +\top; -\top)$

   Sejam $\mathcal{R}_\phi$ e $\mathcal{R}_\varphi$ os DDB que representam $\Omega(\phi)$ e $\Omega(\varphi)$.
2. Usando a relação $\Omega(\phi \vee \varphi) = \Omega(\phi) \cup \Omega(\varphi)$, a fórmula $\phi \vee \varphi$ é representado por $\mathcal{R}_\phi \cup \mathcal{R}_\varphi$.
3. Usando a relação $\Omega(\phi \wedge \varphi) = \Omega(\phi) \cap \Omega(\varphi)$, a fórmula $\phi \wedge \varphi$ é representado por $\mathcal{R}_\phi \cup \mathcal{R}_\varphi$.
4. A negação $\neg\phi$ é representada pelo complemento que corresponde a uma simples troca de sinal do DBB (com a subsequente redução e normalização.
5. A implicação $\phi \supset \varphi$ é considerada equivalente a $\neg\phi \vee \varphi$ e é tratada da mesma forma usando a disjunção e o complemento.

Desta forma pode-se construir o DDB que representa uma fórmula directamente (sem ter de passar por uma fase de conversão da fórmula em formas normais conjuntivas ou disjuntivas).

exemplo: Considere-se a fórmula completamente genérica

$$a \wedge (b \supset \neg(a \vee c)) \vee (a \supset c)$$

que, claramente, não está normalizada. O mecanismo de construção é duplamente recursivo: existe a recursividade na estrutura da fórmula, usando os casos atrás descritos, e depois existe a recursividade para efectuar as uniões ou intersecções de DBB's (como foi descrito na pag. 35).

Neste caso, constrói-se indutivamente os DDB que representam $a \wedge (b \supset \neg(a \vee c))$ e $(a \supset c)$, e depois procede-se à união das DDB's.

Para construir os DDB parcelares procede-se de forma indutiva: selecciona-se o operador principal (neste caso $\wedge$ e $\supset$), calculam-se os DDB dos argumentos e combina-se essas representações segundo a regra atrás definida para o operador em causa.

1.6. Exemplos de problemas SAT

Um problema recorrente todos os anos é a alocação dos professores do ensino básico e secundário na escolas, de acordo com preferências expressas pelos professores, o número de vagas nas escolas, a classificação dos professores e algumas outras regras adicionais.

De facto o problema, tal como o reconhecemos, cai dentro de uma categoria mais vasta de **problemas de alocação**. Por isso a versão que iremos apresentar é ligeiramente diferente deste.

1. Existem n professores (identificados pelos índices $1..n$) e m escolas (identificadas pelos índices $1..m$). A ordem dos índices indica uma classificação absoluta tanto dos professores como das escolas.
   A variável a_{ij} indica se o professor i está alocado à escola j.
2. Cada professor manifesta a sua preferência pelas escolas: P_{ij} indica se o professor i tem interesse na a escola j e $P_i \doteq \{\, j \mid P_{ij} = 1 \,\}$
3. As escolas manifestam a sua preferência nos professores: Q_{ji} indica se a escola j tem interesse no professor i e $Q_j \doteq \{\, i \mid Q_{ji} = 1 \,\}$.

Pretende-se atribuir valores lógicos às variáveis a_{ij} sujeito às regras:

1. Cada escola $j \in 1..m$ preenche uma única vaga com um professor no qual manifestou interesse. Cada professor $i \in 1..n$ ocupa uma única vaga numa escola na qual manifestou interesse.

   É possível generalizar o problema supondo várias vagas numa escola e, até, várias vagas num professor. No entanto este caso reduz-se ao problema como aqui está apresentado: por exemplo, uma escola com N vagas pode ser vista como N escolas, cada uma com uma só vaga.

2. Se o professor i está alocado à escola j então:
   (a) qualquer outro professor $i' < i$, que tenha manisfestado interesse na escola j, tem de estar colocado numa escola j' com $j' < j$.
   (b) qualquer outra escola $j' < j$, que tenha manifestado interesse no professor i, tem colocar um professor i' com $i' < i$.

O problema pode ser generalizado considerando, em vez de uma ordenação única tanto para professores coma para escolas, ordenações esecíficas de cada actor (professor ou escola).

Assim, poderíamos definir **listas de preferências**: uma lista de escolas para cada professor e uma lista de professores para cada escola.

Neste caso, as regras (2) seriam bastante mais complexas; por exemplo, a regra de preferência dos professores seria:

> *se o professor i está colocado na escola j então qualquer professor i', que na lista desta escola esteja melhor colocado que i, tem de estar colocado numa escola j' que, na lista de i', esteja antes de j.*

Vamos tentar descrever este problema como um problema de verificação proposicional.

1. **dados do problema:** são as ordenações iniciais de professores e escolas e as matrizes booleanas P e Q, que traduzem as preferências de uns e outros.
2. **incógnitas:** são os valores booleanos a atribuir às variáveis booleanas a_{ij} de modo a verificar as regras. As variáveis a_{ij} serão, portanto, os nossos símbolos proposicionais.
3. **regras da alocação:** para todo $i \in 1..n$ e todo $j \in 1..m$
   (a) $\bigvee_{(k \in P_i)} a_{ik}$
   cada professor é alocado a uma escola em que mostrou interesse
   (b) $\bigvee_{(k \in Q_j)} a_{kj}$
   cada escola tem alocada um professor em que mostrou interesse
   (c) Para todo $k \neq j$, com $k, j \in P_i$, $\neg a_{ij} \vee \neg a_{ik}$
   nenhum professor pode estar alocado a duas escolas diferentes
   (d) Para todo $k \neq i$, com $k, i \in Q_j$, $\neg a_{ij} \vee \neg a_{kj}$
   nenhuma escola tem alocados dois professores diferentes

4. **regras de preferência:** para todo $i \in 1..n$ e todo $j \in 1..m$
   (a) Para todo $s < i \wedge j \in P_i \cap P_s$ $\neg a_{ij} \vee \bigvee_{(k<j, k \in P_s)} a_{sk}$
   (b) Para todo $r < j \wedge i \in Q_j \cap Q_r$ $\neg a_{ij} \vee \bigvee_{(k<i, k \in Q_r)} a_{kr}$

6 DEFINIÇÃO *Representamos por* $\mathcal{A} \equiv [a_{ij}]$ *a matriz* $n \times m$ *que tem os símbolos proposicionais* a_{ij} *como elementos. Uma* **alocação** α, *para* $\mathcal{A}$, *é um modelo para a lógica proposicional gerada por estes símbolos e escrito na forma de uma matriz booleana* $\alpha \in \mathbb{B}^{n \times m}$.

Exemplo 1: Considere-se uma situação 3×3 (três professores e três escolas) descritos pelas matrizes

$$P = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad Q^T = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

O nosso problema tem 9 variáveis proposicioais $a_{11}..a_{33}$ e as clausulas disjuntivas seguintes

1. *regras de alocação*
   (a) linhas de P: $(a_{11} \vee a_{12})$, $(a_{21} \vee a_{23})$, $(a_{31} \vee a_{32} \vee a_{33})$
   (b) linhas de Q: $(a_{21} \vee a_{31})$, $(a_{12} \vee a_{22} \vee a_{32})$, $(a_{13} \vee a_{23})$
   (c) exclusão nas linhas:

$$\neg a_{11} \vee \neg a_{12} \ , \ \neg a_{11} \vee \neg a_{13} \ , \ \neg a_{12} \vee \neg a_{13}$$

$$\neg a_{21} \vee \neg a_{22} \ , \ \neg a_{21} \vee \neg a_{23} \ , \ \neg a_{22} \vee \neg a_{23}$$

$$\neg a_{31} \vee \neg a_{32} \ , \ \neg a_{31} \vee \neg a_{33} \ , \ \neg a_{32} \vee \neg a_{33}$$

   (d) exclusão nas colunas:

$$\neg a_{11} \vee \neg a_{21} \ , \ \neg a_{11} \vee \neg a_{31} \ , \ \neg a_{21} \vee \neg a_{31}$$

$$\neg a_{12} \vee \neg a_{22} \ , \ \neg a_{12} \vee \neg a_{32} \ , \ \neg a_{22} \vee \neg a_{32}$$

$$\neg a_{13} \vee \neg a_{23} \ , \ \neg a_{13} \vee \neg a_{33} \ , \ \neg a_{23} \vee \neg a_{33}$$

2. *regras de preferênca*
   (a) Para $i = 2, s = 1, j = 1$ tem-se $\neg a_{21} \vee \bigvee_{k<1} \ldots \equiv \neg a_{21}$.
   Para $i = 3, s = 1$ pode-mos ter $j = 1$ ou $j = 2$. Daqui resulta $\neg a_{31} \vee \bigvee_{k<1} \ldots \equiv \neg a_{31}$ e $\neg a_{32} \vee a_{11}$.
   Para $i = 3, s = 2$ pode-mos $j = 1$ e $j = 3$. Daqui resulta $\neg a_{31}$ de novo e ainda $\neg a_{33} \vee a_{21}$.
   (b) Para $j = 2$, $r = 1$ pode-se ter $i = 2$ ou $i = 3$. Do primeiro caso resulta $\neg a_{22}$. Do segundo caso resulta $\neg a_{32} \vee a_{21}$.
   Para $j = 3$, $r = 1$ tem-se apenas $i = 2$ o que origina $\neg a_{23}$.
   Para $j = 3$, $r = 2$ tem-se $i = 1$ e $i = 2$. Daqui resulta $\neg a_{13}$ e $\neg a_{33} \vee a_{12}$.

Em resumo as regras de preferência originam as seguintes clausulas

$$\neg a_{21}\ ,\ \neg a_{31}\ ,\ \neg a_{32} \vee a_{11}\ ,\ \neg a_{33} \vee a_{21} \qquad (9)$$
$$\neg a_{22}\ ,\ \neg a_{32} \vee a_{21}\ ,\ \neg a_{23}\ ,\ \neg a_{13}\ ,\ \neg a_{33} \vee a_{12}$$

Este exemplo constrói as diferentes clausulas directamente das regras. No entanto é simples verificar que são possíveis várias optimizações. Nomeadamente:

1. Se $P_{ij} = 0$ ou $Q_{ji} = 0$ não é possível alocar o professor i à escola j; portanto tem de ser $a_{ij} = 0$. Isto diminui imediatamente o número de incógnitas e também simplifica outras clausulas onde esta variável possa ocorrer.
   No exemplo anterior, a simples observação de P e Q permitia reduzir o número de incógnitas de 9 para 5: a_{12} , a_{21} , a_{23} , a_{31} , a_{32}.
2. Se uma variável aparece sózinha numa linha da matriz $[a_{ij}]$, então, se existir solução, tem de ter o valor 1 e as variáveis que aparecem na mesma coluna tem de ter o valor 0 (já que, em cada linha ou coluna, apenas uma e só uma variável tem valor 1).
   No exemplo anterior isto permite concluir que $a_{12} = 1$ e $a_{32} = 0$.
3. O mesmo argumento se aplica a colunas. No exemplo a variável a_{23} aparece sózinha na última coluna: isto implica $a_{23} = 1$ e , $a_{21} = 0$.
4. Resta a variável a_{31}. Porque a única outra variável na 1ª coluna está a 0, deveria ter o valor 1; a outra única variável na 3ª linha (a_{32}) também está a zero; portanto deve-se ter $a_{31} = 1$.

A análise das matrizes P e Q e o uso das regras da alocação permitiu forçar, neste caso, os valores lógicos a matrix $\mathcal{A} = [a_{ij}]$ e construir uma alocação α que é o único modelo que permite validar todas as clausulas que resultam das regras de alocação.

$$\alpha = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

Porém não é garantida uma solução para o problemas um vez que as regras de precedência não foram testadas. Se determinarmos o valor lógico das

várias clausulas em (9) como avaloração determinada por A verificamos que falham duas das clausulas: $\neg a_{31}$ e $\neg a_{23}$.

Isto significa que este problema, tal como está determinado pelas matrizes P e Q, não tem soluções.

Se o número de de professores não coincidir com o número de escolas não pode existir uma solução. O problema pode ser resolvido introduzindo escolas (se faltarem) ou professores "virtuais" de modo a obter tantas professores como escolas. Uma alocação de uma escola (ou professor) virtual corresponde, efectivamente, a uma "não-alocação".

□

O projecto e segurança de muitas técnicas criptográficas estão ligadas ao estudo das funções da forma $f : \mathbb{B}^n \longrightarrow \mathbb{B}^n$ ou $f : \mathbb{B}^n \longrightarrow \mathbb{B}$ que tomam como argumento uma sequência finita de *bits* de comprimento fixo e devolvem uma sequência do mesmo tamanho ou então um simples bit.

Nas funções booleanas $f : \mathbb{B}^n \longrightarrow \mathbb{B}$ os argumentos x são vectores de n bits; as operações básicas sobre os argumentos são:

1. Selecção $\mathsf{sel} : \mathbb{Z}_n \times \mathbb{B}^n \longrightarrow \mathbb{B}$,

$$\mathsf{sel}(i, x) = x_i$$

   i.e. dado $i \in \mathbb{Z}_n$, seleciona a componente de ordem i do vector x.

2. Operações binárias $\oplus$, $* : \mathbb{B}^n \times \mathbb{B}^n \longrightarrow \mathbb{B}^n$
   são as duas funções binárias que aplicam *xor* e *and* bit a bit.

$$(x \oplus y)_i \;=\; x_i + y_i \qquad (x * y)_i \;=\; x_i \cdot y_i$$

A noção de *índice* pode ser gereralizada; podemos tomar como índice um qualquer conjunto de inteiros $i \in \mathbb{Z}_n$; por exemplo, se tivermos $n = 8$,

um índice será, por exemplo, $\{0, 3, 7\}$. O valor booleano seleccionado por este índice será

$$x_{0,3,7} \doteq x_0 \cdot x_3 \cdot x_7$$

Genéricamente um **índice** para funções boolenas de n argumentos booleanos é um sub-conjunto $u \subseteq \mathbb{Z}_n$; o conjunto desses índices representa-se por $\mathbb{U}_n$ e identifica-se com $\wp(\mathbb{Z}_n)$.

A função selecção $\mathrm{sel} : \mathbb{U}_n \times \mathbb{B}^n \longrightarrow \mathbb{B}$ generaliza-se facilmente

$$\mathrm{sel}(u, x) = \Pi_{i \in u}\, x_i \quad , \quad \mathrm{sel}(\emptyset, x) = 1 \tag{10}$$

7 DEFINIÇÃO *Para* $u \in \mathbb{U}_n$ *designa-se por* x_u *o monómio* $\mathrm{sel}(u, x)$. *Designa-se por* $[x]_u$ *o polinómio* $\mathrm{sel}(u, x) \cdot \Pi_{i \notin u}(1 + x_i)$.

Por exemplo, se for $n = 4$ e $u = \{0, 2\}$, temos

$$x_u = x_0 \cdot x_2 \quad \text{e} \quad [x]_u = x_0 \cdot (1 + x_1) \cdot x_2 \cdot (1 + x_3)$$

Toda a função booleana de domínio $\mathbb{B}^n$ pode ser escrita como um polinómio a n variáveis $x_0, x_1, \ldots, x_{n-1}$ dado por

$$f(x) = \sum_{u \in \mathbb{U}_n} a(u)\, x_u \tag{11}$$

para uma função $a : \mathbb{U}_n \longrightarrow \mathbb{B}$.

Esta é a chamada **forma normal algébrica** de f e é completamente determinada pela função a dita *função de índices* ou *espectro de índices* ou, simplesmente, *espectro* se não existirem ambiguidades.

Uma forma equivalente de representar a mesma função consiste em considerar o conjunto $\mathcal{A} \subseteq \mathbb{U}_n$ de índices u para os quais $a(u) = 1$; essencialmente $\mathcal{A}$ é o conjunto que tem a função de coeficientes como função característica.

Nesse caso (11) escreve-se

$$f(x) \;=\; \sum_{u\in\mathcal{A}} x_u \tag{12}$$

Daqui se deduz que o número total de funções booleanas de argumento $\mathbb{B}^n$ é 2^{2^n}.

O **grau** de f é definido como $(\max_{u\in\mathcal{A}} |u|)$: i.e. a maior cardinalidade de um índice em $\mathcal{A}$.

Se o grau é 0 ou 1 a função diz-se **afim**.

Claramente que o número total de funções afins é 2^{n+1} metade das quais são **lineares**: i.e. funções f tais que $a(\emptyset) = 0$ ou, equivalentemente, $\emptyset \notin \mathcal{A}$.

Exemplo 2: Considere-se as funções booleanas $f : \mathbb{B}^4 \to \mathbb{B}$ com 4 argumentos booleanos. Um exemplo de uma função na forma normal algébrica será

$$f(x) = 1 + x_0 + x_2 + x_1\,x_2 + x_1\,x_3 + x_0\,x_1\,x_3$$

O conjunto dos índices que correspondem a termos não nulos é

$$\mathcal{A} \;=\; \{\emptyset\ ,\ \{0\}\ ,\ \{2\}\ ,\ \{1,2\}\ ,\ \{1,3\}\ ,\ \{0,1,3\}\}$$

O maior deles tem 3 elementos; por isso f tem grau 3.

Esta função não é afim. Um exemplo de função afim será

$$g(x) = 1 + x_0 + x_2$$

que não é uma função linear devido à presença da constante 1. Um exemplo de função linear será

$$h(x) \;=\; x_0 + x_2$$

Em termos de espectros, o de g é $\{\emptyset\ ,\ \{0\}\ ,\ \{2\}\}$ enquanto que o de h é $\{\{0\}\ ,\ \{2\}\}$.

O **símbolo de Kronecker** de ordem n é a função $\delta : \mathbb{B}^n \to \mathbb{B}$ que verifica $\delta(x) = 1$ se e só se $x = (0, 0, \cdots, 0)$.

3 FACTO *Para todo* $x \in \mathbb{B}^n$ *verifica-se*

$$\delta(x) = (1 + x_0) \cdot (1 + x_1) \cdot (1 + x_2) \cdots (1 + x_{n-1}) = [x]_\emptyset$$

Para quaisquer $X, Y \subseteq \mathbb{U}_n$ define-se a sua **união disjunta** por

$$X \uplus Y \doteq (X \setminus Y) \cup (Y \setminus X)$$

e a sua **convolução** por

$$X \circledast Y \doteq \biguplus_{x \in X} \{ x \cup y \mid y \in Y \}$$

4 FACTO *Sejam* $\mathcal{A}, \mathcal{B} \subseteq \mathbb{U}_n$ *os espectros de funções* f, g *respectivamente; i.e.*

$$f(x) = \sum_{u \in \mathcal{A}} x_u \qquad g(x) = \sum_{v \in \mathcal{B}} x_v$$

(i) *Se* f *é uma função constante então: se for* $f(x) = 1$ *, o seu espectro é* $\{\emptyset\}$ *e, se for* $f(x) = 0$ *, o seu espectro é* $\{\ \}$ *.*

(ii) *Se* $\mathcal{A} = \mathbb{U}_n$ *então* f *coincide com o símbolo de Kronecker* δ *. Se* $\mathcal{A} = \mathbb{U}_n \setminus \emptyset$ *então* $f = 1 + \delta$ *(i.e.* $f(x) = 1$ *se e só se* $x \neq 0$*).*

(iii) $(f + g)$ *tem espectro* $\mathcal{A} \uplus \mathcal{B}$ *e* $(f \cdot g)$ *tem espectro* $\mathcal{A} \circledast \mathcal{B}$ *.*

$$(f + g)(x) = \sum_{u \in \mathcal{A} \uplus \mathcal{B}} x_u \qquad (f \cdot g)(x) = \sum_{u \in \mathcal{A} \circledast \mathcal{B}} x_u$$

Corolário:

$1 + f$ *(a negação de* f *) tem espectro* $\mathcal{A} \uplus \emptyset$ *.*

(iv) *Se* $g(x) = f(z * x)$ *, para algum* $z \in \mathbb{B}^n$ *, então*

$$\mathcal{B} = \{ u \in \mathcal{A} \mid z_u = 1 \}$$

O conjunto $f^{-1}(1) = \{ x \mid f(x) = 1 \}$ chama-se **suporte** de f e é representado por $\text{supp}(f)$.

Exemplo 3: Consider-se a função booleana em $\mathbb{B}^4$

$$f(x_0, x_1, x_2, x_3) = 1 + x_0 \cdot x_1 + x_0 \cdot x_2 + x_1 \cdot x_2 \cdot x_3$$

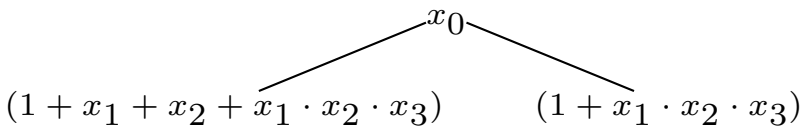
Pode-se sempre escrever

$$f(x_0, x_1, x_2, x_3) \;=\; x_0 \cdot f(1, x_1, x_2, x_3) + (1 + x_0) \cdot f(0, x_1, x_2, x_3)$$

ou seja, com alguma manipulação

$$= x_0 \cdot (1 + x_1 + x_2 + x_1 \cdot x_2 \cdot x_3) + (1 + x_0) \cdot (1 + x_1 \cdot x_2 \cdot x_3)$$

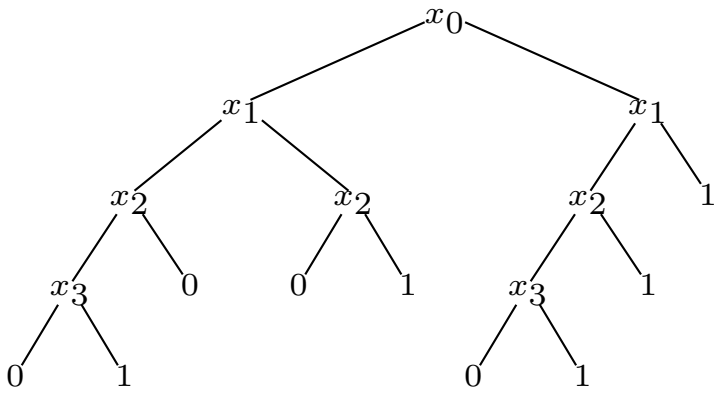
O que sugere uma representação arbórea



Repetindo o processo para as restantes variáveis, vemos que

$$(1 + x_1 + x_2 + x_1 \cdot x_2 \cdot x_3) \;= x_1 \cdot x_2 \cdot (1 + x_3) + (1 + x_1) \cdot (1 + x_2)$$
$$(1 + x_1 \cdot x_2 \cdot x_3) \;= x_1 \cdot (x_2 \cdot (1 + x_3) + (1 + x_2) \cdot 1) + (1 + x_1) \cdot 1$$

O que sugere a seguinte árvore



Note-se que os percursos iniciados na raiz determinam valores de x e os respectivos valores $f(x)$ consoante a folha onde os percursos terminam. Os percursos são determinados pelas regras muito simples: $x_i = 1$ significa "*virar à esquerda* no nodo x_i", enquanto $x_i = 0$ significa "*virar à direita*.no nodo x_i".

Nesta árvore, por exemplo, os seguintes percursos terminam no valor 0

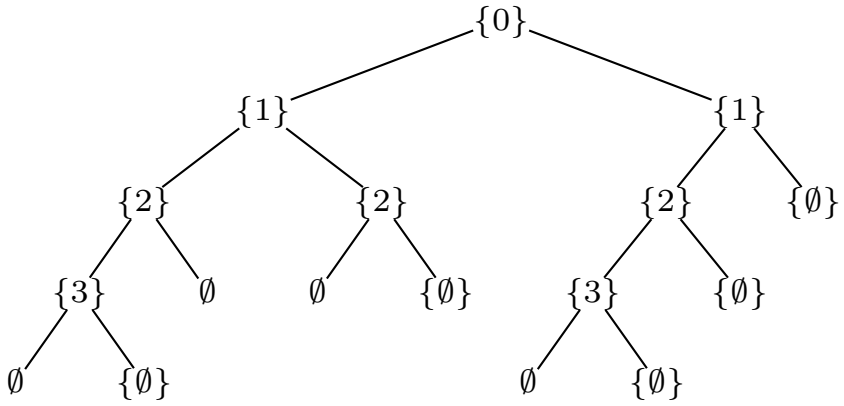
$$\{x_0 = 1\ ,\ x_1 = 0\ ,\ x_2 = 1\ ,\ x_3 = ?\} \quad , \quad \{x_0 = 1\ ,\ x_1 = 1\ ,\ x_2 = 0\ ,\ x_3 = ?\}$$

$$\{x_0 = 1\ ,\ x_1 = 1\ ,\ x_2 = 1\ ,\ x_3 = 1\} \quad , \quad \{x_0 = 0\ ,\ x_1 = 1\ ,\ x_2 = 1\ ,\ x_3 = 1\}$$

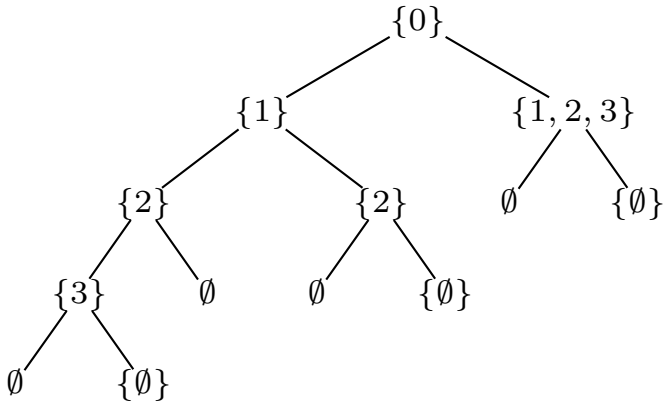
Todos os restantes percursos terminam no valor 1. Isto diz-nos que

$$f(1,0,1,0) = f(1,0,1,1) = f(1,1,0,0) =$$
$$= f(1,1,0,1) = f(1,1,1,1) = f(0,1,1,1) = 0$$

Uma árvore equivalente à anterior pode ser construída colocando nos nodos e nas folhas os espectros das expressões que ocorrem na árvore anterior.



Admitindo que os nodos podem ter quaisquer índices (e não apenas índices singulares) a árvore pode-se simplificar para



Esta árvore permite, agora, reconstruir o espectro da função inicial. De facto é fácil de verificar:

(i) Se a árvore for uma folha o espectro é o que a folha indica: $\emptyset$ ou $\{\emptyset\}$.

(ii) Se a árvore for um triplo $\alpha = \langle\, \sigma\,,\ \alpha^+\,,\ \alpha^-\,\rangle$, o seu espectro $\mathcal{A}$ é

$$\mathcal{A} \;=\; \mathcal{A}^- \uplus \{\sigma\} \pitchfork (\mathcal{A}^+ \uplus \mathcal{A}^-)$$

sendo $\mathcal{A}^+, \mathcal{A}^-$ os espectros representados pelas sub-árvores α^+ e α^-.

Para sistematizar esta construção seja $\mathbb{U}_{k,n} \doteq \wp(\mathbb{Z}_n \setminus \mathbb{Z}_k)$; isto é, o conjunto de todos os índices formado por elementos $k \leq i < n$.

8 DEFINIÇÃO *Dado um polinómio reduzido (sem monómios repetidos)* $f = \sum_{u\in\mathcal{A}} x_u$ *com* $\mathcal{A} \subseteq \mathbb{U}_{k,n}$ *o* **fraccionamento** *de* f *em* $\mathbb{U}_{k,n}$ *é:*

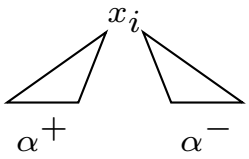
1. *Se* f *é uma função constante, o fraccionamento coincide com a própria função.*
2. *Se* f *não é constante (tem grau maior do que* 0*), sejam:*
   (i) $f^-(x_{k+1}, \cdots, x_{n-1})$ *o polinómio que se obtém eliminando de* f *todos os monómios que contenham* x_k*; seja* α^- *o seu fraccionamento em* $\mathbb{U}_{k+1,n}$ *determinado por este processo.*
   (ii) $f^+(x_{k+1}, \cdots, x_{n-1})$ *o polinómio que se obtém de* f *eliminando* x_k *de todos os seus monómios e reduzindo o resultado final através da eliminação de pares de monómios iguais; seja* α^+ *o seu fraccionamento em* $\mathbb{U}_{k+1,n}$.

   Se $f^- = f^+$*, o fraccionamento* α *de* f *coincide com* $\alpha^+ = \alpha^-$*; em caso contrário, é o triplo* $\alpha = \langle\, x_k\,,\ \alpha^+\,,\ \alpha^-\,\rangle$.

   A **ordem** *do fraccionamento é* 0 *se* f *for constante ou, em caso contrário, é* $(n-i)$ *sendo* i *o índice da variável que constitui o primeiro elemento deste triplo.*

Esta definição sugere imediatamente uma representação arbórea para o fraccionamento de uma função cujo espectro está contido em $\mathbb{U}_{k,n}$. As funções de grau zero serão

as folhas da árvore. As funções não constantes têm fraccionamentos que são triplos $\langle\, x_i\,,\ \alpha^+\,,\ \alpha^-\,\rangle$ (com $i \geq k$) formados por uma variável e dois outros fraccionamentos.

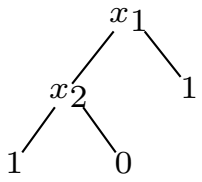


Se a função não for constante existe pelo menos um monómio com uma ou mais variáveis. Não é necessário que contenha a variável x_k; por exemplo, para $k = 0$ e $n = 3$, considere-se a função $f(x_0, x_1, x_2) = 1 + x_1 + x_1 \cdot x_2$.

$f(x_0, x_1, x_2)$ tem 3 monómios nenhum dos quais contém x_0. O cálculo de f^+ e de f^-, com o fraccionamento feito em $\mathbb{U}_{0,3}$, não modifica f uma vez que não contém x_0; teremos, assim, $f = f^+ = f^-$.

Porém, quando se efectua o fraccionamento em $\mathbb{U}_{1,3}$, tem-se $f^- = 1$ e $f^+ = 1 + 1 + x_2$ que, após redução, se simplifica em x_2.

O fraccionamento final de f está representado ao lado e, como vemos, tem ordem 2.



5 FACTO *Sejam f, g duas funções booleanas e α, β os respectivos fraccionamentos em algum $\mathbb{U}_{k,n}$. Representemos por $(\alpha + \beta)$, $(\alpha \cdot \beta)$, $\alpha^{(z)}$ os fraccionamentos, respectivamente, das funções $(f + g)$, $(f \cdot g)$, $f^{(z)}$. Verifica-se:*

1. $(\alpha + 0) = (\alpha \cdot 1) = \alpha$, $(\alpha \cdot 0) = 0$, $0^{(z)} = 0$, $1^{(z)} = 1$, $\alpha \cdot \alpha = \alpha$ *e* $\alpha + \alpha = 0$.
2. *Redução:* $\langle\, x\,,\ \alpha\,,\ \alpha\,\rangle$ *simplifica em* α .
3. *Se for* $\alpha = \langle\, x_k\,,\ \alpha^+\,,\ \alpha^-\,\rangle$ *e* β *é um fracionamento de ordem inferior à de* α*, então*

$$(\alpha + \beta) = \quad \langle\, x_k\,,\ \alpha^+ + \beta\,,\ \alpha^- + \beta\,\rangle$$

$$(\alpha \cdot \beta) = \quad \langle\, x_k\,,\ \alpha^+ \cdot \beta\,,\ \alpha^- \cdot \beta\,\rangle$$

4. *Se* α *e* β *têm a mesma ordem e se for* $\alpha = \langle\, x_k\,,\, \alpha^+\,,\, \alpha^-\,\rangle$ *e* $\beta = \langle\, x_k\,,\, \beta^+\,,\, \beta^-\,\rangle$, *então*

$$\begin{aligned}(\alpha + \beta) = &\quad \langle\, x_k\,,\, \alpha^+ + \beta^+\,,\, \alpha^- + \beta^-\,\rangle \\ (\alpha \cdot \beta) = &\quad \langle\, x_k\,,\, \alpha^+ \cdot \beta^+\,,\, \alpha^- \cdot \beta^-\,\rangle\end{aligned}$$

5. *Se for* $\alpha = \langle\, x_k\,,\, \alpha^+\,,\, \alpha^-\,\rangle$, *então*

$$\alpha^{(z)} = \begin{cases} \langle\, x_k\,,\, (\alpha^+)^{(z)}\,,\, (\alpha^-)^{(z)}\,\rangle & \textit{se} \quad z_k = 0 \\ \langle\, x_k\,,\, (\alpha^-)^{(z)}\,,\, (\alpha^+)^{(z)}\,\rangle & \textit{se} \quad z_k = 1 \end{cases}$$

□

Frequentemente as funções boolenas aparecem agrupadas em vectores. Uma função booleana vectorial $\mathbf{S} : \mathbb{B}^n \longrightarrow \mathbb{B}^n$ pode ser descrita por um vector de n funções booleanas escalares

$$\mathbf{S}(x_1, x_2, \cdots, x_n) = \begin{bmatrix} h_1(x_1, x_2, \cdots, x_n) \\ h_2(x_1, x_2, \cdots, x_n) \\ \cdots \\ h_n(x_1, x_2, \cdots, x_n) \end{bmatrix}$$

Na terminologia criptográfica, uma tal função é designada por uma $n \times n$ **S-Box**.

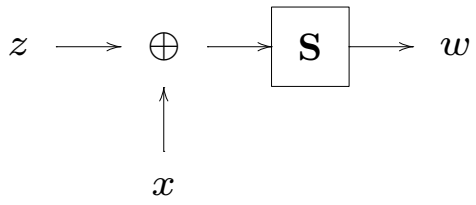
Facilmente se generaliza o conceito para S-Boxes não quadradas: uma $n \times m$ **S-Box** é uma função $\mathbf{S} : \mathbb{B}^n \longrightarrow \mathbb{B}^m$ definida por um vector de m componentes em que, cada uma, é uma função $h_i : \mathbb{B}^n \longrightarrow \mathbb{B}$, com $i \in 0..m-1$.

Exemplo 4: As três funções boolenas

$$\begin{bmatrix} h_0(x) = & 1 + x_0 \cdot x_1 \\ h_1(x) = & x_0 \cdot x_2 \cdot (1 + x_1) \\ h_2(x) = & 1 + x_0 + x_1 + x_2 \end{bmatrix}$$

definem uma SBox quadrada 3×3.

O exemplo mais corrente desta representação pode ser descrito pela figura seguinte



Pode-se ver x como uma chave, z como o texto de uma mensagem a cifrar e w como o criptograma resultante. Alguns problemas:

Dados $z, w \in \mathbb{B}^n$ e **S**,

(I) determinar se existe algum valor de x que seja solução da equação

$$\mathbf{S}(z \oplus x) = w \tag{13}$$

(II) Caso exista gerar aleatoriamente **uma** solução

(III) Caso existam, enumerar **todas** as soluções.

O problema de tipo **I** procura saber, apenas, se existe uma chave, o problema de tipo **II** procura descobrir uma chave e o problema de tipo **III** procura enumerar todas as chaves.

Escrita de outro modo, a equação (13) é $\delta(w \oplus \mathbf{S}(z \oplus x)) = 1$, ou, equivalentemente,

$$\delta^{(w)}(\mathbf{S}^{(z)}(x)) = 1 \qquad \text{ou} \qquad [h^{(z)}]_{\bar{w}} = 1 \tag{14}$$

Exemplo 5: Recuperemos a **SBox** 3×3 do exemplo 4 e suponhamos o seguinte par *entrada-saída*

$$z = (1, 0, 1) \qquad w = (0, 1, 0)$$

A equação que resulta de (14) (com $w_0 = 0, w_1 = 1, w_2 = 0$) será

$$h_0(z \oplus x) \cdot (1 + h_1(z \oplus x)) \cdot h_2(z \oplus x) = 1$$

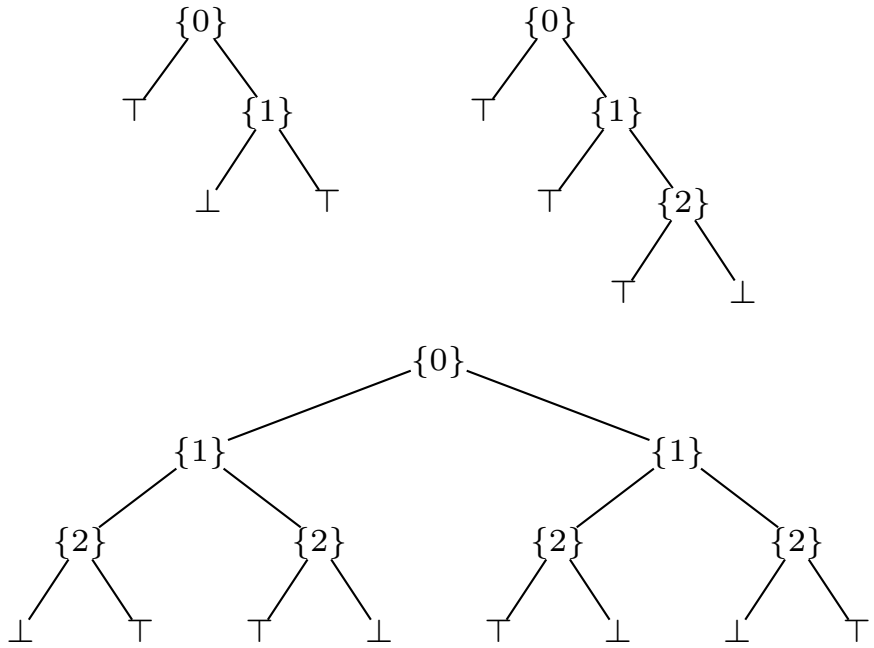
expandindo

$$(1 + (1 + x_0) \cdot x_1) \cdot (1 + (1 + x_0) \cdot (1 + x_1) \cdot (1 + x_2)) \cdot$$
$$\cdot (1 + (1 + x_0) + x_1 + (1 + x_2)) = 1$$

simplificando

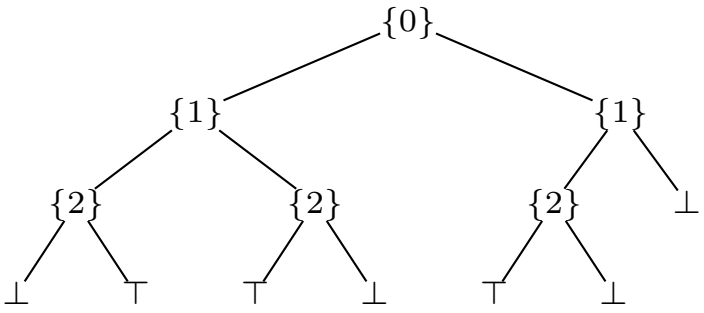
$$(1 + x_1 + x_0 \cdot x_1) \cdot (1 + (1 + x_0) \cdot (1 + x_1) \cdot (1 + x_2)) \cdot (1 + x_0 + x_1 + x_2) = 1$$

Os espectros dos três factores nesta equação serão (abrev. $\top \equiv \{\emptyset\}$, $\bot \equiv \emptyset$),



Usando as regras no facto 5 o espectro resultante será

(15)



2.Lógica de Primeira Ordem

Uma linguagem lógica de **predicados** descreve um determinado universo usando dois tipos de entidades:

1. **termos** que representam os *objectos* do universo de discurso,
2. **fórmulas**, que representam *propriedades* desses objectos.

A construção simbólica destas entidades (os símbolos e as regras de composição de símbolos) constituem a **sintaxe** da linguagem.

A **semântica** da linguagem constrói *significados* para essas entidades,

1. O significados dos termos obtém-se associando termos a elementos de uma determinada estrutura matemática,
2. O significado das fórmulas obtém-se associando cada fórmula a um determinada valor de verdade ou, de forma mais geral, associando-lhe o atributo de ser ou não "válida".

2.1. Sintaxe da Lógica de Predicados

Sintaxe dos Termos

Os objectos de um universo de discurso são determinados por um conjunto $\mathcal{F}$ de **símbolos de função** a cada um dos quais está associado um inteiro positivo designado por **aridade** desse símbolo. Os símbolos com aridade zero são chamados **constantes**.

TERMOS DE BASE: constantes ou entidades da forma $f(t_1, t_2, \cdots, t_n)$ em que $f \in \mathcal{F}$ é um símbolo de aridade $n > 0$ e os diversos $t_1, \cdots, t_n$ são termos de base.
$\mathcal{T}_{\mathcal{F}}$ denota o conjunto dos termos de base gerados por $\mathcal{F}$.

Os objectos de discurso podem ser indeterminados ou conter componentes indeterminadas; para representar essa noção usam-se **variáveis**.

Seja $\mathcal{V}$ um conjunto finito de símbolos de variáveis.

TERMOS: constantes, variáveis ou entidades da forma $f(t_1, t_2, \cdots, t_n)$ em que $f \in \mathcal{F}$ é um símbolo de aridade $n > 0$ e os diversos $t_1, \cdots, t_n$ são outros termos.
$\mathcal{T}_{\mathcal{F},\mathcal{V}}$ denota o conjunto dos termos gerados por $\mathcal{F}$ e por $\mathcal{V}$.

exemplo: Considere-se um universo de discurso em que $\mathcal{F}$ é definido pelos seguintes pares *símbolo-aridade*

$$\mathcal{F} = \{(\text{um}, 0)\ ,\ (\text{zero}, 0)\ ,\ (\text{suc}, 1)\ ,\ (\text{soma}, 2)\}$$

Como exemplos de *termos de base* teremos

$$\text{um}\ ,\ \text{suc(um)}\ ,\ \text{soma(um, suc(um))}\ ,\ \text{soma(soma(zero, um), suc(um))}$$

Nenhum destes termos de base (que também são termos) contém qualquer indeterminação. São as variáveis que representam essa indeterminação.

Considere-se o conjunto de variáveis $\mathcal{V} = \{x, y, z\}$; exemplos de *termos* serão

$$x \ , \ \text{soma}(x, \text{um}) \ , \ \text{soma}(\text{suc}(y), \text{soma}(x, z))$$

Sintaxe das Fórmulas

As propriedades dos objectos do universo de discurso são determinadas por um conjunto finito de **símbolos de predicados** $\mathcal{P}$ a cada um dos quais está associado um inteiro positivo designado por **aridade do predicado**. Os símbolos de aridade 0 são designados por **símbolos proposicionais**.

FÓRMULAS símbolos proposicionais, entidades da forma $p(t_1, \cdots, p_n)$ em que $p \in \mathcal{P}$ é um símbolo de predicado de aridade $n > 0$ e os diversos $t_1, \cdots, t_n$ são termos (ditos **predicados atómicos**), ou então:

(i) entidades da forma $\phi \vee \varphi$, $\phi \supset \varphi$, $\phi \wedge \varphi$, $\neg\phi$, $\bot$, em que ϕ, φ são fórmulas,

(ii) entidades da forma $\forall x.\phi$ e $\exists x.\phi$, sendo ϕ uma fórmula e x uma variável.

exemplo: As propriedades *par* e *ímpar* requerem a introdução de dois *símbolos de predicado*, respectivamente par e ímpar de aridade 1; pode-se também considerar um símbolo maior de aridade 2:

$$\mathcal{P} = \{(\text{par}, 1) \ , \ (\text{ímpar}, 1) @ (\text{maior}, 2)\}$$

Exemplos de propriedades

$$\text{par}(x) \ , \ \text{impar}(\text{soma}(\text{zero}, y)) \ , \ \text{maior}(\text{um}, \text{zero})$$

Outros exemplos

$$\forall x.\, \text{maior}(x, \text{zero}) \ , \ \forall y.\, \text{par}(x) \supset \exists x.\, \text{maior}(x, y)$$

Uma linguagem lógica construída nestes moldes designa-se por **Linguagem de Primeira Ordem**.

É possível pensar numa linguagem que represente não só propriedades de objectos como também propriedades de transformações de objectos, ou propriedades de propriedades, ou propriedades de transformações de propriedades, etc. Uma tal linguagem já não é "*de 1ª ordem*" e designa-se por **lógica de ordem superior**.

2.2. Substituições

Variáveis Livres e Ligadas

A noção de termo e predicado determinado é expresso através da noção de *variável livre* e de *variável ligada* numa fórmula.

Um termo ou um predicado atómico que não contenha variáveis diz-se **fechado** ou **determinado**.

Numa fórmula ϕ uma variável associada a um quantificador diz-se **ligada em** ϕ; uma variável que não está associada a nenhum quantificador diz-se **livre em** ϕ. Um predicado sem variáveis livres diz-se **fechado** ou **determinado**.

exemplo: O seguinte programa PROLOG determina o conjunto de variáveis livres num predicado genérico. Dada a possível ambiguidade na representação de variáveis e constantes como átomos PROLOG, o parâmetro V é usado para definir o conjunto de variáveis em cada predicado PROLOG.

```
% vari\'aveis livres
livres(V,-(P),  U) :- livres(V,P,U).
livres(V,P & Q ,U) :- livres(V,P,U1), livres(V,Q,U2), uniao(U1,U2,U).
livres(V,P v Q ,U) :- livres(V,P,U1), livres(V,Q,U2), uniao(U1,U2,U).
livres(V,P => Q,U) :- livres(V,P,U1), livres(V,Q,U2), uniao(U1,U2,U).
livres(V,qualquer(X,P),U) :- livres(V,P,U1), delete(U1,X,U).
livres(V,existe(X,P) , U) :- livres(V,P,U1), delete(U1,X,U).
livres(V,P,U)      :- P =.. [_|Args], varss(V,[],Args,U).

% vari\'aveis acumuladas em termos
vars(V,U1,T,U)   :- atom(T),
        ((member(T,V),\+ member(T,U1)) ->  U = [T|U1] ; U = U1).
vars(V,U1,T,U)   :- T =.. [_|Args], varss(V,U1,Args,U).
varss(_,U,[],U).
varss(V,U1,[H|R],U) :-  vars(V,U1,H,U2), varss(V,U2,R,U).
```

Exemplos

```
| ?- livres([x,y], p(x) => qualquer(y, q(y,x)), U).
U = [x] ?
yes
| ?- livres([x,y], p(x) => qualquer(x, q(y,x)), U).
U = [x,y] ?
yes
```

Substituição

Para determinar fórmulas indeterminadas (e além do uso de quantificadores) é possível usar *substituições*.

A **substituição num termo** t de uma variável x por um segundo termo μ é uma transformação de termos cujo resultado se escreve

$$t\,[\mu/x] \qquad \text{ou} \qquad t\,[x := \mu]$$

A substituição define-se indutivamente na forma do termo:

(i) Se t coincide com x então $t\,[\mu/x] \equiv \mu$; todos os outros átomos (constantes ou variáveis diferentes de x) não são afectados pela substituição.

(ii) Em alternativa a substituição de t obtém-se aplicando a substituição aos argumentos desse termo.

O seguinte programa PROLOG implementa a substituição em termos; cada substituição $[t/x]$ é representada por um par `[x,t]`.

```
% substituicao em termos
subsT(V,[X,T],Y,U) :- atom(Y), member(X,V),
                      (X == Y -> U=T ; U=Y).
subsT(V,[X,T],W,U) :-
     W=..[F|As], subsTs(V,[X,T],As,Bs), U =..[F|Bs].
subsTs(V,[X,T],Ts,Us) :- map(Ts,subsT(V,[X,T]),Us).
```

$\sim \square \sim$

A **substituição numa fórmula** Φ de uma variável x por um termo μ é uma transformação que só afecta Φ se x for uma variável livre nessa fórmula.

Nesse caso a fórmula transformada, representada por

$$\Phi\,[\mu/x] \qquad \text{ou} \qquad \Phi\,[x := \mu]$$

define-se da mesma forma indutiva usada na substituição em termos:

(i) $\Phi\,[\mu/x] \equiv \Phi$ se x não é livre em Φ.

(ii) Se x é livre em Φ e $\Phi \equiv \mathrm{Op}\,\Phi_1 \ldots \Phi_n$, em que Op é um dos prefixos $\forall y.$ ou $\exists y.$, é um conectivo ou é um símbolo de predicado, então a substituição x em Φ obtém-se aplicando essa mesma substituição a cada um dos elementos $\Phi_1 \ldots \Phi_n$.

O seguinte propgrama PROLOG implementa esta definição em três cláusulas para o predicado subsF: a primeira aplica-se a fórmulas quantificadas, a segunda a fórmulas construídas por conectivos e a terceira a predicados atómicos construídos a partir de termos. Só na 1ª cláusula é necessário verificar se a variável de substituição é livre ou não.

A última cláusula extende a substituição a listas de fórmulas.

```
% substitui\c c\~ao em f\'ormulas
subsF(V,[X,T],W,U) :-
   W=.. [Op,Y,P], (Op= qualquer ; Op= existe),
   (X == Y -> P=Q ; subsF(V,[X,T],P,Q)), U=.. [Op,Y,Q].
subsF(V,[X,T],W,U) :-
   W=.. [Op|As], (Op= '-' ; Op= '&' ; Op= 'v' ; Op= '=>'),
   subsFs(V,[X,T],As,Bs), U=.. [Op|Bs].
subsF(V,[X,T],W,U) :-
   W=.. [Op|As], subsTs(V,[X,T],As,Bs), U=.. [Op|Bs].
subsFs(V,[X,T],Ws,Us) :- map(Ws,subsF(V,[X,T]),Us).
```

Exemplos de aplicação deste programa

```
| ?- subsF([x,y],[x,a], p(x) => qualquer(y,q(x) & p(y)), U).
U = p(a)=>qualquer(y,q(a)&p(y)) ?
yes
| ?- subsF([x,y],[x,a], p(x) => qualquer(x,q(x) & p(y)), U).
U = p(a)=>qualquer(x,q(x)&p(y)) ?
yes
```

2.3. Semântica da Lógica de Predicados

A LPO é formada por duas classes de entidades sintácticas: *termos* $\mathcal{T}$ (representando objectos) e *predicados* $\mathcal{L}$ (representando propriedades); assim dar significado à LPO implica definir o significado de termos e o significado de predicados.

Dar significado a um predicado Φ realiza-se verificando a sua **validade** num determinado modelo $\mathcal{M}$ expressa numa relação

$$\mathcal{M} \models \Phi$$

Dar significado a um termo bem determinado t (i.e. um termo sem variáveis) realiza-se associando esse termo a um objecto matemático bem determinado $\mu \in S$ elemento de um conjunto S previamente escolhido. Uma tal associação $t \mapsto \mu$ designa-se por **interpretação** do termo t na estrutura S e é descrita por uma função

$$\sigma : \mathcal{T} \longrightarrow S$$

Exemplo

Os termos 1, $1+1-1$ e $0+1$ são termos bem determinados, distintos entre si, que podem ser todos interpretados em $\mathbb{N}$ pela unidade 1.

Esta interpretação dá o mesmo significado aos três termos. Note-se, porém, que esta a interpretação $\mathcal{T} \rightarrow \mathbb{N}$, é apenas uma das interpretações possíveis; outras interpretações darão significados distintos aos três termos.

Nomeadamente é possível tomarmos para estrutura $S \equiv \mathcal{T}$ e, como função de interpretação σ, a função identidade; isto é, cada termo é interpretado por si próprio e dois termos distintos são sempre interpretados de modo diferente.

$$\sim \Box \sim$$

A interpretação dada por $S \equiv \mathcal{T}$ e $\sigma \equiv \mathrm{id}$ chama-se **interpretação de Herbrand** e os modelos que daí resultam chamam-se **modelos de Herbrand**.

A interpretação de Herbrand só define significado de termos determinados. Para um termo t com variáveis é necessário definir valores para as várias variáveis que podem ocorrer nesse termo.

A associação de valores a variáveis é feita pela noção de *atribuição*: isto é, uma função que associa variáveis a elementos da estrutura de interpretação S. Na interpretação de Herbrand $S \equiv \mathcal{T}$; portanto são termos; logo

Numa interpretação de Herbrand uma **atribuição** é uma função

$$\upsilon \ : \ \mathcal{V} \longrightarrow \mathcal{T}$$

Dado um termo t representamos por t^{υ} o termo determinado que se obtém substituindo cada variável x pelo seu valor $\upsilon(x)$

$$t^{\upsilon} \ = \ t\,[\upsilon(x)/x] \qquad \text{para todo} \quad x \in \mathcal{V}$$

exemplo: O programa PROLOG seguinte implementa atribuições como listas de pares `[x,t]`. Por exemplo a atribuição

$$\{\, x \mapsto a \ , \ y \mapsto b \ , \ z \mapsto c(a,b) \,\}$$

é representada pela lista `[[x,a],[y,b],[z,c(a,b)]]`.

O programa implementa a substituição múltipla através do "predicado de ordem superior" `foldr`.

```
% Substitui\c c\~ao m\'ultipla
subsFF(V,Atrib,W,U)  :- foldr(W,Atrib,subsF(V),U).
subsTT(V,Atrib,W,U)  :- foldr(W,Atrib,subsT(V),U).
```

Exemplo

```
| ?- subsTT([x,y,z],[[x,a],[y,b],[z,c(a,b)]], t(x,y,z), U).
U = t(a,b,c(a,b)) ?
yes
```

2.4. Modelos de Herbrand na LPO

Um **modelo de Herbrand** é um conjunto, finito ou não, de predicados atómicos determinados; i.e. predicados da forma $\mathsf{p}(t_1, \cdots, t_n)$ em que os termos t_i não contêm variáveis.

Exemplo 6: Numa linguagem com símbolos de predicado $\mathcal{P} \equiv \{ >, \mathsf{par} \}$ e constantes $\mathcal{C} \equiv \{ 0, 1, 2 \}$, o seguinte conjunto é um modelo de Herbrand,

$$\mathcal{M} \equiv \{ 2 > 1 \ , \ 1 > 0 \ , \ \mathsf{par}(0) \ , \ \mathsf{par}(2) \}$$

A definição de validade de predicados tem de considerar separadamente predicados determinados e predicados indeterminados; predicados indeterminados são válidos ou não consoante os valores eventualmente associados às suas variáveis livres.

Tal como nos termos a associação de variáveis a valores é feita por *atribuições* $\upsilon : \mathcal{V} \rightarrow \mathcal{T}$. Dado um predicado indeterminado Φ, define-se

$$\Phi^{\upsilon} \equiv \Phi\,[\upsilon(x)/x] \qquad \text{para todo} \quad x \in \mathcal{V} \tag{16}$$

Para um predicado determinado Φ, a validade exprime-se numa relação binária entre um modelo $\mathcal{M}$ e Φ escrita

$$\mathcal{M} \models \Phi$$

Se Φ é indeterminado a validade exprime-se numa relação ternária entre um modelo $\mathcal{M}$, uma atribuição υ e o predicado Φ, escrita

$$\mathcal{M}, \upsilon \models \Phi$$

e definida, a partir da relação binária, por

$$\mathcal{M}, \upsilon \models \Phi \qquad \text{sse} \qquad \mathcal{M} \models \Phi^{\upsilon} \tag{17}$$

A relação binária $\models$ para predicados determinados, atómicos ou gerados por conectivos, define-se indutivamente da forma usual:

- $\mathcal{M} \not\models \bot$
- $\mathcal{M} \models \mathsf{p}(t_1 \ldots t_n)$ sse $\mathsf{p}(t_1 \ldots t_n) \in \mathcal{M}$
- $\mathcal{M} \models \Phi \wedge \Psi$ sse $\mathcal{M} \models \Phi$ e $\mathcal{M} \models \Psi$
- $\mathcal{M} \models \Phi \vee \Psi$ sse $\mathcal{M} \models \Phi$ ou $\mathcal{M} \models \Psi$
- $\mathcal{M} \models \Phi \supset \Psi$ sse $\mathcal{M} \not\models \Phi$ ou $\mathcal{M} \models \Psi$

Para predicados quantificados determinados, em modelos de Herbrand, a sua validade pode ser definida percorrendo todas as eventuais substituições da variável de quantificação

- $\mathcal{M} \models (\forall x.\, \Phi)$ sse $\mathcal{M} \models \Phi\,[t/x]$ para todo $t \in \mathcal{T}$
- $\mathcal{M} \models (\exists x.\, \Phi)$ sse $\mathcal{M} \models \Phi\,[t/x]$ para algum $t \in \mathcal{T}$

$$\sim \Box \sim$$

A noção de validade aplica-se a teorias Γ

- Em *teorias fechadas* (nenhum $A \in \Gamma$ contém variáveis livres).
  $\mathcal{M} \models \Gamma$ se $\mathcal{M} \models A$ para todo $A \in \Gamma$
- Em *teorias abertas* (com variáveis livres):
  $\mathcal{M}, \upsilon \models \Gamma$ se $\mathcal{M}, \upsilon \models A$ para todo $A \in \Gamma$

e permite definir as noções de **consequência semântica**, **tautologia** e **inconsistência** percorrendo os vários modelos $\mathcal{M}$ e atribuições υ.

- $\Gamma \models \Phi$ se $\mathcal{M}, \upsilon \models \Gamma$ implica $\mathcal{M}, \upsilon \models \Phi$ para todo $\mathcal{M}$ e todo υ.
- Φ é uma *tautologia* se $\mathcal{M}, \upsilon \models \Phi$ para todo $\mathcal{M}$ e todo υ.
- Γ é *inconsistente* se $\mathcal{M}, \upsilon \not\models \Gamma$ para todo $\mathcal{M}$ e todo υ.

A noção de consequência semântica e de consistência de uma teoria são essenciais à correcção de sistemas de dedução na LPO.

Esta primeira proposição estabelece alguns resultados de consistência que são independentes da forma específica das fórmulas.

9 PROPOSIÇÃO *Seja* Γ *uma teoria e* Φ *um fórmula*

(a) *Verifica-se* $\Gamma \models \Phi$ *se e só se* $\{ \Gamma \ , \ \neg\Phi \}$ *é inconsistente.*
(b) *A teoria vazia* $\Gamma = \emptyset$ *nunca é inconsistente.*
(c) *Se* $\bot \in \Gamma$ *então* Γ *é inconsistente.*
(d) Γ *é inconsistente se e só se* Γ^{υ} *é inconsistente para todo* υ.

Face à proposição 9, o resultados (a), (b) e (c) seguintes estabelecem, respectivamente, regras de *inclusão*, *monotonia* e *corte*.

10 PROPOSIÇÃO *Nas condições da proposição anterior*

(a) *Toda a teoria* Γ *que contenha* Φ *e* $\neg\Phi$ *é inconsistente.*
(b) *Se* Γ *é inconsistente e* $\Delta \supseteq \Gamma$ *então* Δ *é inconsistente.*
(c) *Se* $\{ \Gamma \ , \neg\Phi \}$ *e* $\{ \Gamma \ , \Phi \}$ *são inconsistentes, então* Γ *é inconsistente.*

A componente proposicional da LPO exprime-se da forma seguinte:

11 PROPOSIÇÃO *Sejam* $\boldsymbol{\alpha}$ *e* $\boldsymbol{\beta}$ *predicados genéricos conjuntivos e disjuntivos definidos nos seguintes quadros.*

$\boldsymbol{\alpha}$	$\boldsymbol{\alpha_1}$	$\boldsymbol{\alpha_2}$
$P \wedge Q$	P	Q
$\neg(P \vee Q)$	$\neg P$	$\neg Q$
$\neg(P \supset Q)$	P	$\neg Q$

$\boldsymbol{\beta}$	$\boldsymbol{\beta_1}$	$\boldsymbol{\beta_2}$
$\neg(P \wedge Q)$	$\neg P$	$\neg Q$
$P \vee Q$	P	Q
$P \supset Q$	$\neg P$	Q

Então

(a) $\{ \Gamma \ , \boldsymbol{\alpha} \}$ *é inconsistente se e só se* $\{ \Gamma \ , \boldsymbol{\alpha_1} \ , \boldsymbol{\alpha_2} \}$ *é inconsistente.*

(b) $\{\,\Gamma\ ,\ \boldsymbol{\beta}\,\}$ *é inconsistente se e só se* $\{\,\Gamma\ ,\ \boldsymbol{\beta_1}\,\}$ *e* $\{\,\Gamma\ ,\ \boldsymbol{\beta_2}\,\}$ *são inconsistentes.*

(c) $\{\,\Gamma\ ,\ \neg\neg\Phi\,\}$ *é inconsistente se e só se* $\{\,\Gamma\ ,\ \Phi\,\}$ *é inconsistente.*

As provas de ambos as proposições são uma consequência imediata das definições do acetato anterior.

4 TEOREMA *Seja* $t \in \mathcal{T}$ *um termo fechado e* a *uma qualquer variável que não ocorre em* Γ *ou* Φ *.*

(a) Se a teoria $\{\,\Gamma\ ,\ (\forall x.\,\Phi)\ ,\ \Phi[t/x]\,\}$ *é inconsistente então também a teoria* $\{\,\Gamma\ ,\ (\forall x.\,\Phi)\,\}$ *é inconsistente.*

(b) Se a teoria $\{\,\Gamma\ ,\ (\exists x.\,\Phi)\ ,\ \Phi[\mathsf{a}/x]\,\}$ *é inconsistente então também a teoria* $\{\,\Gamma\ ,\ (\exists x.\,\Phi)\,\}$ *é inconsistente.*

(c) Se a teoria $\{\,\Gamma\ ,\ \neg(\exists x.\,\Phi)\ ,\ \neg\Phi[t/x]\,\}$ *é inconsistente então também* $\{\,\Gamma\ ,\ \neg(\exists x.\,\Phi)\,\}$ *é inconsistente.*

(d) Se a teoria $\{\,\Gamma\ ,\ \neg(\forall x.\,\Phi)\ ,\ \neg\Phi[\mathsf{a}/x]\,\}$ *é inconsistente então também* $\{\,\Gamma\ ,\ \neg(\forall x.\,\Phi)\,\}$ *é inconsistente.*

Prova A prova formal deste importante teorema pode ser vista no texto de *M. Fitting* (citado como referência complementar deste curso). Um esquema de prova pode, no entanto ser apresentado para dois dos resultados:

Em (a), se $\{\Gamma, (\forall x.\,\Phi)\}$ fosse válido para algum modelo $\mathcal{M}$ e atribuição υ então seria $\mathcal{M}, \upsilon \models \Phi[t/x]$ contrariando a inconsistência de $\{\Gamma, (\forall x.\,\Phi), \Phi[t/x]\}$.

Em (b), se $\{\Gamma, (\exists x.\,\Phi)\}$ fosse válido num par $\mathcal{M}, \upsilon$ existiria um termo t tal que $\mathcal{M}, \upsilon \models \Phi[t/x]$. Definindo uma nova atribuição $\upsilon' = \upsilon \cup \{\mathsf{a} \mapsto t\}$, é possível construir um par $\mathcal{M}, \upsilon'$ que valida todas as fórmulas em Γ, Φ que o anterior par validava (o que é possível porque a não ocorre em Γ, Φ) e ainda valida $\Phi[\mathsf{a}/x]$. Portanto

$\{\Gamma, (\exists x.\, \Phi), \Phi[a/x]\}$ não poderia ser inconsistente o que contraria a nossa hipótese.

Os resultados (c) e (d) provam-se de forma análoga a (a) e a (b).

$\sim \Box \sim$

Note-se que, em (b), a "*nova*" variável a pode ser vista como um *identificador* para o valor específico t onde Φ é válido. Para não criar ambiguidades com nomes existentes tem de ser diferente dos outros identificadores que ocorrem em Γ, Φ. Esta abordagem é seguida na definição de sistemas de dedução baseados neste resultado.

Do mesmo modo que fórmulas proposicionais de agrupam em fórmulas $\boldsymbol{\alpha}$ (conjuntivas) e fórmulas $\boldsymbol{\beta}$ (disjuntivas), também as fórmulas quantificadas se podem agrupar em fórmulas $\boldsymbol{\delta}$, ditas *universais*, e fórmulas $\boldsymbol{\gamma}$, ditas *existenciais*.

Se acordo com os seguintes quadros, para cada termo termo fechado t, uma fórmula $\boldsymbol{\delta}$ está associado a uma fórmula $\boldsymbol{\delta}(t)$ e, para cada variável $\mathbf{a}$, uma fórmula $\boldsymbol{\gamma}$ está associado a uma fórmula $\boldsymbol{\gamma}(\mathbf{a})$.

$\boldsymbol{\delta}$	$\boldsymbol{\delta}(t)$
$\forall x.\, \Phi$	$\Phi[t/x]$
$\neg\exists x.\, \Phi$	$\neg\Phi[t/x]$

$\boldsymbol{\gamma}$	$\boldsymbol{\gamma}(\mathbf{a})$
$\exists x.\, \Phi$	$\Phi[\mathbf{a}/x]$
$\neg\forall x.\, \Phi$	$\neg\Phi[\mathbf{a}/x]$

2 COROLÁRIO *Seja $t \in \mathcal{T}$ um qualquer termo fechado e a uma qualquer variável que não ocorre em Γ ou $\boldsymbol{\gamma}$.*

(a) $\{\, \Gamma\ ,\ \boldsymbol{\delta}\ ,\ \boldsymbol{\delta}(t)\, \}$ *é inconsistente se e só se* $\{\, \Gamma\ ,\ \boldsymbol{\delta}\, \}$ *é inconsistente.*

(b) $\{\, \Gamma\ ,\ \boldsymbol{\gamma}(\mathbf{a})\, \}$ *é inconsistente se e só se* $\{\, \Gamma\ ,\ \boldsymbol{\gamma}\, \}$ *é inconsistente.*

Prova

Para as fórmulas $\boldsymbol{\delta}$ a proposição 9 diz-nos que a inconsistência de

$\{ \Gamma \ , \ \boldsymbol{\delta} \}$ implica a inconsistência de $\{ \Gamma \ , \ \boldsymbol{\delta} \ , \ \boldsymbol{\delta}(t) \}$. O teorema 4 determina o resultado inverso: a inconsistência da segunda teoria implica a inconsistência da primeira.

Para fórmulas $\boldsymbol{\gamma}$, sendo $\{ \Gamma \ , \ \boldsymbol{\gamma}(\mathbf{a}) \}$ inconsistente também será inconsistente $\{ \Gamma \ , \ \boldsymbol{\gamma} \ , \ \boldsymbol{\gamma}(\mathbf{a}) \}$ (proposição 9) e, pelo teorema 4, é inconsistente $\{ \Gamma \ , \ \boldsymbol{\gamma} \}$.

Inversamente, sendo $\{ \Gamma \ , \ \boldsymbol{\gamma} \}$ inconsistente, assumindo que $\{ \Gamma \ , \ \boldsymbol{\gamma}(\mathbf{a}) \}$ não é inconsistente obtem-se uma contradição.

Tomemos, por exemplo, o caso $\boldsymbol{\gamma} \equiv \exists x. \ \Phi$. Se $\{\Gamma, \Phi[\mathbf{a}/x]\}$ não for inconsistente existitirá um modelo $\mathcal{M}$ e uma atribuição υ que valida Γ e $\Phi[\mathbf{a}/x]$. Usando a "testemunha" $t \equiv \upsilon(\mathbf{a})$, esse modelo validará também $\exists x. \ \Phi$ contrariando a hipótese de inconsistência de $\{\Gamma, \exists x. \ \Phi \}$.

2.5. Unificação e Resolução

Quase todos os sistemas dedutivos para a Lógica de Primeira Ordem são baseados na resposta a um tipo de questões

> Dada uma teoria Γ será que ela é inconsistente?

Na LPO Clássica sabe-se que não é possível construir um algoritmo que, dada uma teoria genérica Γ, seja capaz de dar uma resposta a esta questão num número finito de passos. Ou seja, a LPO clássica é **indecidível**.

Porém algumas sub-linguagens da LPO são decidíveis; nomeadamente a chamada **linguagem de clausulas**.

9 Definição

Um ***literal*** *em LPO é um predicado atómico* $p(t_1, \ldots t_n)$ *ou a sua negação* $\neg p(t_1, \ldots t_n)$.

Uma ***clausula aberta*** *é uma fórmula do tipo*

$$\phi_1(x_1, \ldots, x_n) \vee \phi_2(x_1, \ldots, x_n) \vee \ldots \vee \phi_k(x_1, \ldots, x_n)$$

em que os diversos ϕ_i *são literais e os* x_i *denotam a totalidade das variáveis livres que neles ocorrem. A* ***clausula nula*** $\perp$ *é a clausula determinada pela disjunção de um conjunto vazio de literais.*

Uma ***clausula fechada*** *é uma fórmula fechada do tipo* $\forall x_1 x_2 \ldots x_n \, \Phi$, *sendo* Φ *uma clausula aberta. Uma* ***clausula de base*** *é uma clausula fechada sem quantificadores.*

As ***teorias*** *na linguagem de clausulas são conjuntos finitos de clausulas fechadas.*

Note-se que:

1. Nos literais não ocorrem quaisquer conectivos (para além do eventual $\neg$) ou quantificadores. Podem existir variáveis livres; isto é, os predicados atómicos não são necessariamente fórmulas de base nem os seus argumentos são termos de base.
2. Cada clausula aberta é uma disjunção de literais; portanto, para além da disjunção $\vee$ e de, eventualmente, a negação, não ocorrem quaisquer outros conectivos ou quantificadores.

   Todas as variáveis que ocorrem nos literais são variáveis livres na clausula aberta porque ela não contém qualquer quantificador que "feche" variáveis.
3. Uma clausula fechada $\forall x_1 x_2 \ldots x_n \; \Phi$, é uma fórmula fechada; portanto não contém quaisquer variáveis livres; isto significa que todas a variável livre em Φ é fechada por um dos quantificadores universais $\forall x_i$.
4. A clausula nula $\bot$ e a teoria vazia (sem clausulas) fornecem formas triviais de determinar inconsistência; de acordo com a proposição 9 toda a teoria que contenha $\bot$ é inconsistente e a teoria $\{\ \}$ é sempre consistente.

exemplo: Numa linguagem com símbolos de predicados (todos unários) $\mathbf{p}, \mathbf{q}$, símbolos de função f, g (ambos unários), constantes $0, 1$ e variáveis X, Y, Z temos, como exemplos de clausulas abertas

$$\mathbf{p}(X) \vee \neg\mathbf{q}(Y) \vee \mathbf{q}(f(X)) \quad , \quad \mathbf{q}(g(X)) \vee \neg\mathbf{p}(Y)$$

Como clausulas fechadas teremos

$$\forall X \, \forall Y \; \mathbf{p}(X) \vee \neg\mathbf{q}(Y) \vee \mathbf{q}(f(X)) \quad , \quad \forall X \; \mathbf{p}(f(0)) \vee \mathbf{q}(g(X))$$

Clausulas de base não contém quaisquer variáveis ou quantificadores; por exemplo,

$$\mathbf{p}(f(0)) \vee \mathbf{q}(g(1))$$

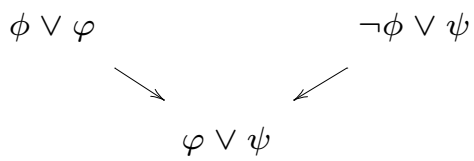
O teorema 4 e as proposições 9 e 11 fornecem a base para definir um algoritmo que permita determinar quando é que uma teoria formada exclusivamente por clausulas é inconsistente.

Numa primeira aproximação, vamos considerar clausulas de base.

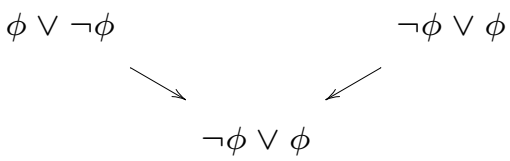
10 DEFINIÇÃO *Duas clausulas de base da forma* $\Phi = \phi \vee \varphi$ *e* $\Psi = \neg\phi \vee \psi$ *, em que* ϕ *é um predicado atómico, dizem-se* ***resolúveis sobre*** ϕ *.*

A clausula $\varphi \vee \psi$ *designa-se por* ***resolvente*** *de* Φ *e* Ψ *.*

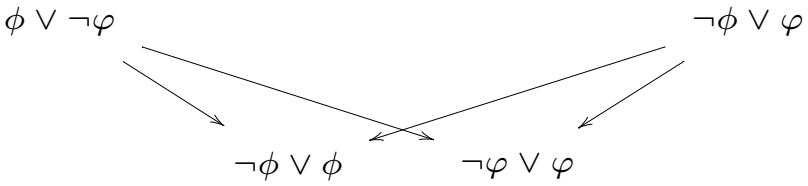
É vulgar representar esta construção por um diagrama da forma



Note-se que duas clausulas de base resolúveis não são necessariamente distintas nem o resolvente é único. Por exemplo a clausula $\phi \vee \neg\phi$ é resolúvel consigo própria; o resolvente é a própria clausula.



O par de clausulas $\phi \vee \neg\varphi$ e $\neg\phi \vee \varphi$ é resolúvel de duas formas distintas produzindo os resolventes $\phi \vee \neg\phi$ e $\varphi \vee \neg\varphi$.



12 PROPOSIÇÃO *Sejam* $\Phi = \phi \vee \varphi$ *e* $\Psi = \neg\phi \vee \psi$ *duas clausulas de base resolúveis. Então toda a teoria* Γ *que contenha* Φ *e* Ψ *é inconsistente se e só se é inconsistente a teoria* $\{\Gamma\ ,\ \varphi \vee \psi\}$ *que se obtém de* Γ *juntando-lhe o resolvente.*

Prova Expandindo as disjunções como se indica na proposição 11, vemos que Γ será inconsistente se e só se são inconsistentes todas as teorias

$$\{\Gamma\ ,\ \phi\ ,\ \neg\phi\} \qquad \{\Gamma\ ,\ \varphi\} \qquad \{\Gamma\ ,\ \psi\}$$

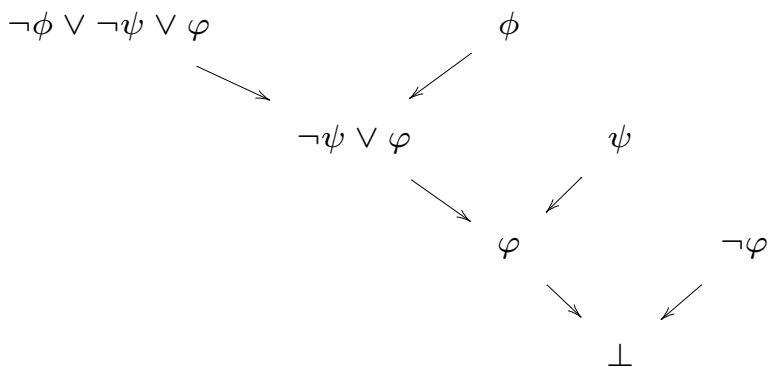
A primeira é inconsistente; as duas últimas são simultaneamente inconsistentes se e só se for inconsistente $\{\Gamma, \varphi \vee \psi\}$.

É possível usar este resultado para provar a inconsistência de uma teoria formada apenas por clausulas de base.

exemplo: Por exemplo, se for

$$\Gamma = \{\neg\phi \vee \neg\psi \vee \varphi \ , \ \phi \ , \ \psi \ , \ \neg\varphi\}$$

constrói-se a seguinte **grafo de resolventes**.



Os vários nodos desta árvore são as fórmulas adicionadas à teoria inicial Γ preservando o seu *status* de consistência.

No final introduziu-se a clausula vazia $\perp$ como resolvente do par de clausulas φ e $\neg\varphi$. Obtém-se deste modo uma teoria que, por conter $\perp$, é inconsistente e que, pelo processo de resolução, tem o mesmo *status* que a teoria original.

Consequentemente concluimos que Γ é inconsistente.

13 PROPOSIÇÃO *Seja Γ uma teoria e ϕ um literal de base que é* ***puro*** *em relação a Γ – i.e. a sua negação não ocorre em nenhuma clausula de Γ. Então Γ é inconsistente se e só se é inconsistente a teoria que se obtém de retirando-lhe todas as clausulas de base que contém ϕ.*

Prova (*sugestão*) Começamos por ver que Γ e $\{\Gamma, \phi\}$ têm o mesmo *status* de consistência.

Por simplicidade vamos supor que ϕ é um predicado atómico (a prova seria idêntica se fosse a negação de um predicado atómico).

Se Γ for inconsistente então $\{\Gamma, \phi\}$ será necessariamente inconsistente. Inversamente, se $\{\Gamma, \phi\}$ fosse inconsistente e Γ não fosse então existiria um modelo $\mathcal{M}$ que não continha ϕ mas validava Γ; acrescentando ϕ a $\mathcal{M}$ obtém-se um novo modelo $\mathcal{M}' = \{\phi\} \cup \mathcal{M}$ que continua a validar Γ, já que $\neg\phi$ não ocorre em Γ, e passa agora a validar ϕ.

Isto contraria a assunção de que $\{\Gamma, \phi\}$ é inconsistente.

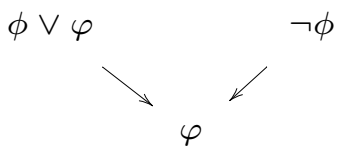
Para qualquer clausula $\phi \vee \Phi$ temos que $\{\Gamma, \phi \vee \Phi\}$ é inconsistente se e só se forem inconsistente $\{\Gamma, \phi\}$ e $\{\Gamma, \Phi\}$. Acabamos de ver que o primeiro é inconsistente se e só se Γ o for; sabemos também que a inconsistência de Γ implica a inconsistência de $\{\Gamma, \Phi\}$. Logo $\{\Gamma, \phi \vee \Phi\}$ é inconsistente se e só se Γ for inconsistente.

Este resultado avança na direcção contrária à resolução: vai retirando clausulas da teoria com o objectivo de, se possível, se chegar à teoria vazia. Essa teoria será necessariamente consistente.

exemplo:

Na teoria $\Gamma = \{\phi \vee \varphi, \neg\phi\}$ as fórmulas ϕ e φ denotam predicados atómicos de base.

De início é possível construir um único resolvente.



Temos uma teoria com uma clausula pura φ. O resultado anterior diz-nos que podemos eliminar todas as clausulas que contém φ; assim é possível retirar φ e $\phi \vee \varphi$.

Resta uma teoria $\Gamma' = \{\neg\phi\}$. Agora $\neg\phi$ é uma clausula pura que, usando de novo o resultado anterior, pode ser eliminada. Resta a teoria vazia $\{\ \}$ que é consistente.

Todo este algoritmo mantém o *status* das várias teorias que terminam numa teoria consistente. Logo a teoria inicial é consistente.

Para localizar os literais puros numa teoria basta usar o seguinte critério:

ϕ é puro em Γ se e só se não existe nenhum par de clausulas desta teoria que sejam resolúveis sobre ϕ.

A essência deste estudo é a Lógica de Primeira Ordem; por isso ele deve incidir essencialmente sobre teorias com clausulas quantificadas.

O *status* de consistência de teorias com clausulas quantificadas baseia-se no resultado apresentado no teorema 4; recordando

Para todo o termo de base t o *status* de consistência da teoria $\{\Gamma\ ,\forall x.\Phi\}$ é o da teoria $\{\Gamma\ ,\ \forall x.\ \Phi\ ,\ \Phi[t/x]\}$

O cálculo de resolventes serviu, nas teorias só com clausulas de base, para determinar os respectivos *status* de consistência. Obviamente que esse cálculo era aí facilitado pelo facto de, com um conjunto finito de clausulas de base, é também finito o número de clausulas geradas por resolução.

Em clausulas que envolvem variáveis e quantificadores já a resolução (como veremos em seguida) tem capacidade para gerar novas clausulas em número contável mas infinito. Um algoritmo que tire conclusões sobre resolventes pode não terminar.

11 DEFINIÇÃO *Sejam* Φ *e* Φ' *duas clausulas abertas que não partilham variáveis. Diz-se que:*

1. Φ **instância** Φ', *e escreve-se* $\Phi \geq \Phi'$ *quando existe uma substituição* σ *tal que* $\Phi' = \sigma(\Phi)$.
2. *Diz-se que* Φ **subsume** Φ', *e escreve-se* $\Phi \supseteq \Phi'$, *quando* Φ *instância um número de literais de* Φ'.

   Formalmente quando, para uma clausula δ *e uma substituição* σ, *se escreve* $\Phi' = \sigma(\Phi) \vee \delta$.
3. Φ **unifica com** Φ', *e escreve-se* $\Phi \asymp \Phi'$, *quando partilham uma instância comum. A substituição que produz a clausula mais geral que é*

instância de Φ *e de* Φ' *chama-se* ***unificador mais geral*** *(abreviado para,* ***umg****) de* Φ *e de* Φ' *.*

4. $\Phi = \phi \vee \Psi$ *e* $\Phi' = \neg\phi' \vee \Psi'$ *são* ***resolúveis sobre*** *o par de literais* ϕ, ϕ' *, se se verifica* $\phi \asymp \phi'$ *.*
   Seja σ *o unificador mais geral do par* ϕ, ϕ' *; o* ***resolvente*** *de* Φ *e* Φ' *é a clausula*

$$\Phi \times \Phi' \doteq \sigma(\Psi) \vee \sigma(\Psi')$$

$$\phi \vee \Psi \xrightarrow{\sigma} \sigma(\Psi) \vee \sigma(\Psi') \qquad \neg\phi' \vee \Psi' \xrightarrow{\sigma} \sigma(\Psi) \vee \sigma(\Psi')$$

$$\sigma(\phi) = \sigma(\phi')$$

A relação de instância determina uma ordem parcial desde que sejam consideradas equivalentes clausulas que se distinguem apenas nos nomes das variáveis.

Com esta relação de ordem, uma clausula unificadora U de Φ e Φ' verifica $U \leq \Phi$ e $U \leq \Phi'$.

Nas clausulas de base, um par de clausulas resolúvel tinha de conter o par ϕ e $\neg\phi$ (um em cada clausula). Nas clausulas abertas a resolução exige que as clausulas contenham um par $\phi, \neg\phi'$, em que ϕ pode ser distinto de ϕ' desde que unifiquem.

Por exemplo, considere-se o par de clausulas (as variáveis em clausulas distintas são sempre distintas)

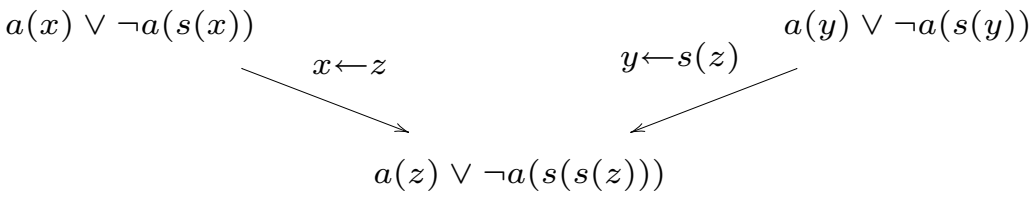
$$\{a(s(x)) \vee b(x) \ , \ \neg a(y) \vee c(y)\}$$

O unificador mais geral de $a(y), a(s(x))$ é $[s(x)/y]$; a resolução será

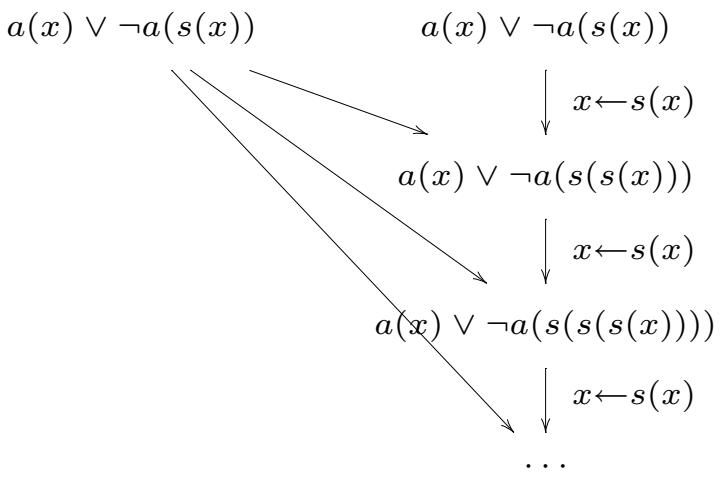
$$a(s(x)) \vee b(x) \xrightarrow{x \leftarrow x} b(x) \vee c(s(x)) \qquad \neg a(y) \vee c(y) \xrightarrow{y \leftarrow s(x)} b(x) \vee c(s(x))$$

Um segundo exemplo mais complexo resulta da observação de que a clausula $a(x) \vee \neg a(s(x))$ é resolúvel consigo própria.

Escrevendo a clausula duas vezes, com variáveis diferentes, temos



Mas o resolvente volta a ser resolúvel com a clausula inicial; pode-se calcular um novo resolvente, que é de novo resolúvel com a clausula inicial, etc.



Esta clausula, por resolução, gera um conjunto infinito de clausulas.

Seja $\Phi = \Phi(x_1, \ldots, x_n)$ uma clausula aberta; representamos por $[\Phi]$ (o **fecho** de Φ) a clausula fechada $\forall x_1 \cdots x_n \, \Phi(x_1, \ldots, x_n)$.

5 TEOREMA *Se Γ é uma teoria que contém um par de clausulas fechadas $[\Phi]$ e $[\Phi']$ tais que Φ e Φ' são resolúveis, é inconsistente se e só se for inconsistente*

$$\{\Gamma \ , \ [\Phi \times \Phi']\}$$

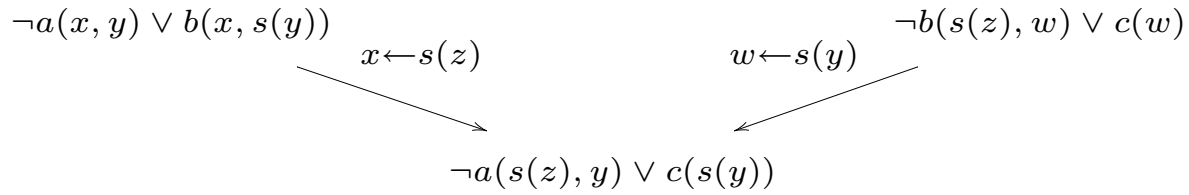
É importante notar-se que a resolução é calculada nas clausulas abertas apesar de a teoria ser formada por clausulas fechadas.

Isto permite renomear as variáveis de modo que não exista conflito nos nomes das variáveis em Φ e Φ'.

Por exemplo, uma teoria

$$\Gamma = \{\forall x\, y.\, \neg a(x,y) \vee b(x, s(y)) \quad , \quad \forall x\, y.\, \neg b(s(x), y) \vee c(y)\}$$

formada por duas clausulas fechadas, dá origem a duas clausulas abertas onde o nome das variáveis pode ser convenientemente escolhido de forma a não existirem ambiguidades.



O resolvente das clausulas abertas é $\neg a(s(z), y) \vee c(s(y))$, com z, y como variáveis livres. O seu fecho será a fórmula

$$\forall z\, y.\, \neg a(s(z), y) \vee c(s(y))$$

A partir deste resultado básico, com as limitações que são introduzidas pela unificação de clausulas abertas (que podem conduzir, como vimos, a teorias infinitas) todo o processo de dedução baseado em resolventes que vimos para as clausulas de base pode ser usado nas clausulas fechadas mais gerais.