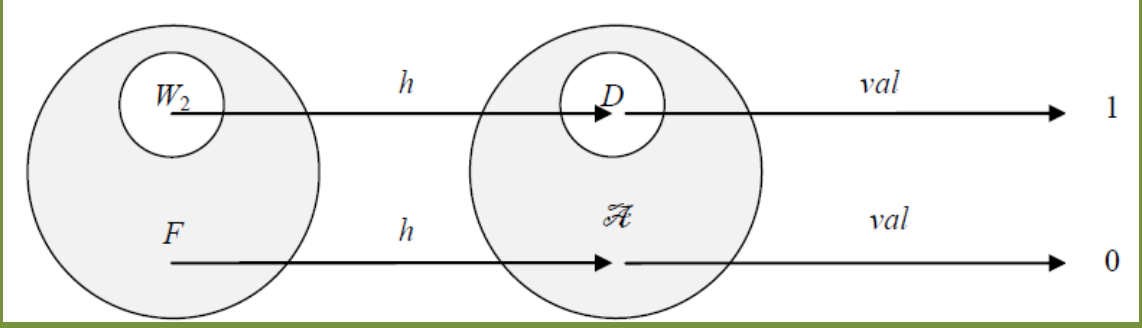
Luís M. Augusto

Lógicas multivalentes

Uma introdução matemática e computacional



http://luismaugusto.my-free.website

1ª versão do manuscrito publicada *online*

Manuscrito submetido para publicação

Para citar esta versão:

Luís M. Augusto (2016). *Lógicas multivalentes. Uma introdução matemática e computacional.* http://luismaugusto.my-free.website

To all the curious, born or otherwise

Este texto é uma adaptação de uma tese de mestrado realizada na Universidade Aberta em 2014 sob a (co-)orientação de Mário Jorge Edmundo e Reinhard Kahle. Além destes, os meus agradecimentos vão também para Carlos Caleiro, membro do júri.

Conteúdo

Lista de Figuras

0 Introdução

0.1 Motivação e objetivos

Por *lógica formal* ou *simbólica* entende-se tipicamente a *lógica clássica*. Esta tem sido um utensílio indispensável desde os finais do séc. XIX na matemática quer pura quer aplicada. Esta última sobretudo, concretizada em grande medida no computador digital, motivou contudo o desenvolvimento de uma panóplia de "novas" lógicas. Com efeito, verificou-se que certos problemas postos pelas novas tecnologias no que diz respeito a acurácia, segurança, certeza, etc., não são tratáveis do ponto de vista da lógica clássica, exigindo outros sistemas lógicos capazes de formalizar aspetos tão díspares como a incompletude ou a inconsistência da informação disponível, a incerteza no raciocínio, requisitos temporais e modais, etc. Estas lógicas, quer apenas estendendo a lógica clássica quer substituindo-a em aplicações *ad hoc*, são hoje conhecidas como *lógicas não clássicas*.

Entre estas, as *lógicas multivalentes*, cuja característica distintiva é o facto de possuírem mais do que os dois valores de verdade ditos clássicos (verdadeiro e falso), são particularmente importantes para aplicações-chave; contam-se, entre estas, por exemplo a análise de circuitos, a programação em lógica, a verificação de *hardware*, o raciocínio não monotónico, a análise de comunicação com *feedback* e o processamento de linguagem natural. As lógicas multivalentes comummente generalizam a lógica clássica, o que faz delas utensílios fundamentais para investigar aspetos centrais de sistemas lógicos clássicos tais como as relações entre a teoria da demonstração e a semântica; em particular, estas lógicas mostraram ter importantes aplicações na investigação das relações entre a lógica clássica e a lógica intuicionista. De facto, e num sentido abstrato – considerando-se as classes de equivalência obtidas a partir de uma noção específica de equivalência lógica como valores de verdade – todo o sistema lógico pode ser visto como uma lógica multivalente.

Para que um sistema lógico (ou uma lógica) possa ser útil tem de satisfazer requisitos de tratabilidade computacional. Dois componentes são especialmente importantes para este fim: um sistema lógico deve possuir, primeiramente, uma semântica simples e intuitiva, e ainda um sistema de demonstração analítico correspondente

para estratégias de demonstração eficazes, nomeadamente no que respeita à automatização da dedução. A *dedução automática*, ou *demonstração automática de teoremas*, é hoje um requisito-chave em qualquer lógica, uma vez que as estratégias de dedução podem ser laboriosas e conter erros, em especial quando não se pode evitar níveis de alta complexidade. A automatização da dedução em lógica clássica – quer proposicional quer de primeira ordem – está já bastante desenvolvida e há hoje muitos demonstradores automáticos disponíveis. Contudo, o terreno das lógicas não clássicas só recentemente se tornou um objeto para a automatização da dedução e mostra-se muito desigualmente desbravado, com muito por investigar e fazer.

A *resolução* é um procedimento de decisão por refutação baseado no *problema da satisfazibilidade* para o qual se desenvolveram já demonstradores automáticos bastante eficientes (ex.: Prover9, Gandalf, Vampire). Este procedimento estende-se naturalmente às lógicas multivalentes por meio da *lógica assinalada*, um formalismo para expressar conetores e quantificadores multivalentes. A lógica assinalada permite-nos raciocinar "classicamente" em lógicas multivalentes, o que representa a aplicação nestas dos já bem conhecidos resultados obtidos na dedução automática em lógica clássica. Por esta razão, tratamos a automatização da demonstração de teoremas em lógicas multivalentes por meio da *resolução assinalada*, ou seja, a resolução implementada em lógica assinalada. Outras técnicas de automatização da demonstração de teoremas em lógicas multivalentes são abordadas noutros textos dedicados a esta temática indicados na Bibliografia.

0.2 Metodologia

Dada a vasta panóplia de sistemas lógicos multivalentes foi necessário proceder a uma seleção dos mesmos. A seleção levada a cabo teve como critérios primeiramente a importância dos vários sistemas a nível tanto teórico como prático e, em segundo lugar, a facilidade com que se prestam à automatização por meio da resolução assinalada. Aborda-se aqui as lógicas multivalentes de um ponto de vista axiomático e algébrico. A axiomatização destas lógicas em sistemas formais adequados não só mostra o seu estádio e estatuto de formalização no panorama da lógica matemática contemporânea, mas pode ser ainda útil para muitas aplicações práticas. No que respeita à abordagem algébrica, restringimos a nossa discussão aos aspetos mais relevantes, como por exemplo a importante noção de matriz lógica.

Dos sistemas lógicos multivalentes selecionados, os sistemas de Łukasiewicz, em especial $Ł_3$ e $qŁ_3$, são aqueles com um tratamento mais profundo dadas a sua importância quer histórica quer contemporânea e a "naturalidade" com que se deixam

traduzir para lógica assinalada. Dos sistemas finitamente multivalentes, os outros cálculos proposicionais apresentadas na secção 2.4 podem ser de igual modo traduzidos para lógica assinalada; na secção 2.5 fornecemos indicações gerais para a tradução de outros sistemas multivalentes quantificados. Quanto aos sistemas infinitamente multivalentes, nomeadamente as chamadas lógicas difusas, a sua importância atual torna a sua abordagem obrigatória numa obra como esta. Apesar dos problemas em aberto em termos de axiomatização, alguns fragmentos destas lógicas são tradutíveis para lógica assinalada, permitindo assim a aplicação neles do cálculo de resolução assinalado. Assim sendo, discutimos algumas das propostas nesse sentido que nos parecem mais viáveis.

O tratamento do cálculo de resolução, nomeadamente da resolução assinalada para lógicas multivalentes, é aqui efetuado tendo o problema da satisfazibilidade (abreviando: problema SAT) como pano de fundo, pelo que são abordados os aspetos principais deste problema que caracteriza em grande medida a lógica matemática contemporânea e em especial a teoria da computabilidade, particularmente no que diz respeito ao fundamental problema da decisão.

Esta obra inova em termos de seleção de aspetos centrais de uma vastíssima literatura que nem sempre é coerente quer na notação quer nos resultados. Inova ainda na organização do material apresentado. Muito deste material é já considerado "folclore" da lógica matemática, pelo que neste caso não se procede à identificação de fontes; estas são indicadas sempre que tal se considerou relevante. O material selecionado, folclore ou outro, foi ainda submetido a alterações em favor da simplificação e compreensibilidade, com algumas definições, proposicões e demonstrações originais ou alvo de reformulações substanciais.

No estudo da lógica é fundamental a resolução de exercícios e integramos neste texto os exercícios que nos parecem mais adequados. Este texto não é uma introdução à lógica clássica, mais não apresentando que os aspetos desta que são essenciais para a abordagem das lógicas multivalentes. Assim sendo, parte-se do princípio que o leitor tem conhecimentos básicos de lógica simbólica. Os exercícios apresentados neste contexto são de mera revisão e remete-se o leitor para bibliografia apresentada ao longo do texto para uma revisão ou um aprofundamento de aspetos da lógica clássica. Faz-se notar que um sólido conhecimento da lógica clássica, tanto no que respeita à teoria da demonstração como à teoria dos modelos, é fundamental para a compreensão das lógicas multivalentes uma vez que, como se disse, estas são comummente generalizações daquela.

De igual modo, o conhecimento do cálculo de resolução para a lógica clássica é essencial para uma compreensão do cálculo de resolução assinalada para as lógicas

multivalentes. Assim sendo, dedicou-se um capítulo inteiramente ao cálculo de resolução (ver Capítulo 3). Um outro objetivo deste capítulo é o de familiarizar o leitor com um demonstrador automático de teoremas baseado na resolução, neste caso o Prover9/Mace4. Um objetivo mais geral é o de convidar o leitor a refletir sobre possíveis modos de utilizar este (ou outro) demonstrador automático na demonstração de teoremas das lógicas multivalentes, levando assim o leitor a participar ativamente num tópico de investigação contemporânea em lógica matemática e computacional.

Não se pressupõe por parte do leitor conhecimentos matemáticos para além das noções básicas da teoria dos conjuntos. Com isto em mente e de modo a permitir a todos os leitores uma compreensão integral deste texto providenciou-se um tratamento sucinto de várias noções matemáticas em caixas imediatamente abaixo dos parágrafos em que tais noções são primeiramente mencionadas.

1 Noções preliminares: Metateoria, definições básicas e notação

A *lógica* é a ciência do raciocínio, e a *lógica dedutiva* é o estudo do raciocínio válido. Embora a lógica não se reduza à lógica dedutiva, havendo ainda (pelo menos) as lógicas ditas *indutiva* e *abdutiva*, por "lógica" entende-se comummente lógica dedutiva. A *lógica matemática (dedutiva)* (ou ainda: *lógica simbólica* ou *formal*) nada mais é do que a circunscrição do estudo da lógica a um contexto matemático. Embora este trabalho se debruce sobre lógica matemática, omitiremos o adjetivo "matemático" (ainda: "simbólico"), a menos que o seu uso seja relevante.

O *objeto* imediato da lógica é constituído por (classes de) *sistemas lógicos* e correspondentes *lógicas*. A noção central da lógica é a *consequência* ou *derivabilidade lógica*. Com efeito, um *sistema lógico* S é um par $(\mathcal{L}, Cq)$, em que $\mathcal{L}$ é uma *linguagem lógica* e Cq é uma *operação de consequência*. A lógica estuda pois tanto aquilo que faz com que uma linguagem ou um *sistema simbólico* ou *formal* seja um sistema lógico, ou seja, um sistema caracterizado pela consequência lógica, como o modo como ele se comporta do ponto de vista da *inferência* (dedutiva, indutiva, ou abdutiva). Assim, a lógica contém uma *metateoria* bem como uma *teoria das técnicas* que explicam a consequência lógica. A primeira debruça-se sobre resultados fundamentais que têm a ver com a consistência, a completude, etc., de um sistema lógico; a segunda pode ser de dois tipos, conhecidos como *teoria da demonstração*, se as técnicas respeitam a consequência *sintática*, e a *teoria de modelos* quando se considera a consequência *semântica*. (As duas noções de consequência coincidem no caso de completude (forte), como se verá.) Começamos por desenvolver os aspetos metateóricos básicos mais relevantes.

1.1 Linguagem lógica

1.1.1 A linguagem $\mathcal{L}$

1.1.1.1 Linguagem-objeto e metalinguagem

Num sentido restrito, a lógica é a interpretação, do ponto de vista da noção de consequência lógica, de expressões bem formadas de acordo com regras estipuladas numa linguagem-objeto; a estas expressões bem formadas chamaremos simplesmente *fórmulas* (abreviando "fórmulas bem formadas"), uma vez que expressões mal formadas não são fórmulas. Uma *linguagem-objeto* é a linguagem na qual certas "coisas" se demonstram, enquanto uma *metalinguagem* é a linguagem na qual se conduz o estudo de uma linguagem-objeto. Neste texto, a metalinguagem é o Português complementado com um vocabulário técnico (ex.: conetor) e com símbolos (ex.: $\vdash$) que não fazem parte das linguagens-objeto abordadas. De um ponto de vista formal, uma linguagem-objeto é o conjunto das sequências de símbolos que obedecem a regras específicas de formação estipuladas na gramática ou sintaxe dessa linguagem.[1] As definições que se seguem enquadram de um modo geral o estudo técnico da linguagem-objeto que denotamos por $\mathcal{L}$.

1.1.1.2 Sintaxe e semântica

1.1.1. DEFINIÇÃO. O *alfabeto* de uma linguagem-objeto $\mathcal{L}$ consiste em (1) um suplemento infinito de *símbolos* para *variáveis de objeto*, bem como para *predicados* e *funções* de aridade $n \geq 0$; (2) um número finito de (símbolos para) *operadores* $\bigstar_1, ..., \bigstar_r$ e *quantificadores* $\blacklozenge_1, ..., \blacklozenge_k$. As *constantes* são símbolos para funções de aridade 0. Os parênteses são tratados como sinais de pontuação.

Tipicamente, (1) são símbolos do alfabeto latino, com ou sem subíndice, e (2) são símbolos convencionais especiais.

1.1.2. DEFINIÇÃO. Dado um alfabeto finito Σ, um *átomo* A é uma expressão da forma $P(t_1, ..., t_n)$ em que P é um símbolo que denota um predicado de aridade n e $t_1, ..., t_n$, os argumentos de A (denotado por $\arg(A)$), são *termos* construídos a partir de símbolos de variáveis e de funções (e logo de constantes) pertencentes a

[1] De um ponto de vista mais restrito – adotado comummente em programação – uma linguagem L é simplesmente a sintaxe de L.

Σ. Um átomo é uma *fórmula*. Se $\bigstar_i$ é um operador e $\phi_1, ..., \phi_n$ são fórmulas, então $\bigstar_i(\phi_1, ..., \phi_n)$ é uma fórmula. $\blacklozenge_i x(\phi)$, em que x é uma variável, também é uma fórmula.

Tipicamente, letras do alfabeto grego representam expressões arbitrárias (ou seja, termos e fórmulas) na metalinguagem.

1.1.3. DEFINIÇÃO. A cada operador $\bigstar_i$ e quantificador $\blacklozenge_i$ de $\mathcal{L}$ encontram-se associadas, para $0 < i \leq n$,

- uma *tabela de verdade* $\widetilde{\bigstar}_i : W^n \longrightarrow W$, e
- uma *função de verdade* $\widetilde{\blacklozenge}_i : \left(2^W - \emptyset\right) \longrightarrow W$

em que $W = \{v_0, v_1, ..., v_{n-1}\}$ é o conjunto de *valores de verdade*.

1.1.4. DEFINIÇÃO. Uma base (*frame*) para uma linguagem lógica $\mathcal{L}$ com um alfabeto Σ, $\mathcal{L}_\Sigma$, e W é um par $(\mathscr{D}, \Theta)$ em que $\mathscr{D}$ é um *domínio de discurso* não vazio e Θ é uma *interpretação de assinatura*, ou seja, um mapeamento das funções $\mathscr{D}^n \longrightarrow \mathscr{D}$ e $\mathscr{D}^n \longrightarrow W$ para cada símbolo de função de aridade n e cada símbolo de predicado de aridade n do alfabeto Σ, respetivamente.

1.1.5. DEFINIÇÃO. Uma *interpretação* $\mathcal{I}$ para $\mathcal{L}_\Sigma$ é um triplo $(\mathscr{D}, \Theta, \delta)$ em que $(\mathscr{D}, \Theta)$ é uma base e δ é uma *atribuição de variáveis* $\delta : V \longrightarrow \mathscr{D}$. Dizemos que $\mathcal{I}$ se baseia em $(\mathscr{D}, \Theta)$.

1.1.6. DEFINIÇÃO. Para $\mathcal{I} = (\mathscr{D}, \Theta, \delta)$, $\mathcal{I}$ é uma interpretação para $\mathcal{L}_\Sigma$ e W, existe uma *função de valoração* correspondente $val_\mathcal{I}$ que se define indutivamente do seguinte modo:

- $val_\mathcal{I}(x) = \delta(x)$ para todo x em Σ.
- $val_\mathcal{I}(f(t_1, ..., t_n)) = \Theta(f)(val_\mathcal{I}(t_1), ..., val_\mathcal{I}(t_n))$ para todo o símbolo de função f de aridade $n \geq 0$ em Σ.
- $val_\mathcal{I}(P(t_1, ..., t_n)) = \Theta(P)(val_\mathcal{I}(t_1), ..., val_\mathcal{I}(t_n))$ para todo o símbolo de predicado P de aridade $n \geq 0$ em Σ.

- $val_{\mathcal{I}}\left(\bigstar_i\left(\phi_1,...,\phi_n\right)\right)=\widetilde{\bigstar}_i\left(val_{\mathcal{I}}\left(\phi_1\right),...,val_{\mathcal{I}}\left(\phi_n\right)\right)$ para todo o operador lógico $\bigstar_i$ em Σ.

- $val_{\mathcal{I}}\left(\left(\blacklozenge_i x\right)\phi\right)=\widetilde{\blacklozenge}_i\left(distr_{\mathcal{I},x}\left(\phi\right)\right)$ para todo o quantificador $\blacklozenge_i$ em Σ, em que $distr_{\mathcal{I},x}\left(\phi\right)=\left\{val_{\mathcal{I}_d^x}\left(\phi\right)\mid d\in\mathscr{D}\right\}$ é a *distribuição* de ϕ em $\mathcal{I}$ com respeito a x e $\mathcal{I}_d^x$ é a interpretação idêntica a $\mathcal{I}$ uma vez dado $\delta(x)=d$.

As definições 1.1.1 e 1.1.2 estabelecem os elementos básicos da *sintaxe* de $\mathcal{L}$ (abreviando$\mathcal{L}_\Sigma$); as definições 1.1.3 - 1.1.6 estabelecem uma *semântica* para $\mathcal{L}$.

1.1.2 Ordem de uma lógica

Uma lógica apresenta-se em *ordens* de acordo com as definições seguintes:

1.1.7. DEFINIÇÃO (ordem de um predicado). Um predicado tem ordem 1 se todos os seus argumentos são termos; caso contrário, tem ordem $n+1$, para n a ordem superior dos seus argumentos.

1.1.8. DEFINIÇÃO (ordem de um quantificador). Um quantificador tem ordem 1 se quantifica uma variável individual; caso contrário tem ordem $n+1$, para n a ordem do predicado (ou função) quantificado.

1.1.9. DEFINIÇÃO (ordem de uma fórmula). A ordem de uma fórmula é a ordem superior de um dos seus quantificadores e predicados.

1.1.10. DEFINIÇÃO (ordem de uma lógica). Uma lógica de ordem n é uma lógica cujas fórmulas têm ordem n ou inferior. Falamos assim de lógica de ordem zero para $n<1$, lógica de primeira (segunda) ordem para $n=1$ ($n=2$, respetivamente) e lógica de ordem superior para $n>2$.

Quando a ordem é 0, ou seja, quando os predicados têm aridade 0 (i.e, são variáveis proposicionais), falamos de *lógica proposicional* (LP) ou ainda de *cálculo proposicional* (CPp); quando a ordem é 1, temos uma *linguagem* ou *lógica de primeira ordem* (LPO) ou ainda um *cálculo de predicados* (CPr).

1.1.3 As linguagens $\mathcal{L}^{Prop}$ e $\mathcal{L}^{Pred}$

Começamos por definir a linguagem de CPp e em seguida aumentamo-la de modo a obter CPr.

1.1.11. DEFINIÇÃO. Uma *linguagem proposicional* $\mathcal{L}^{Prop}$ é um par ordenado (Vp, O) em que $Vp = \{p, q, r, p_1, q_1, r_1, ...\}$ é um conjunto enumerável de *variáveis proposicionais* e $O = \{O_1, ..., O_n\}$ é um conjunto enumerável de operadores com aridade $n \geq 1$. As variáveis proposicionais chamam-se *átomos proposicionais* (ainda: *fórmulas atómicas*), e os operadores chamam-se *conetores lógicos*.

Denotamos por F o conjunto de fórmulas de uma linguagem proposicional.

1.1.12. DEFINIÇÃO. Uma *expressão* ou *fórmula bem formada* (*fbf*) de $\mathcal{L}^{Prop}$ define-se indutivamente do seguinte modo:

- todo o átomo proposicional é uma fbf, ou seja, $Vp \subseteq F$;
- se $\phi_1, ..., \phi_n$ são fórmulas e O_i é um conetor lógico com aridade n, então $O_i(\phi_1, ..., \phi_n)$ é uma fbf. (Usaremos notação infixa. Como se disse acima, uma fbf é simplesmente uma *fórmula*.)

1.1.13. DEFINIÇÃO. Uma *matriz lógica* $\mathfrak{M}$ para uma linguagem $\mathcal{L}^{Prop} = (Vp, O)$ é um triplo $\mathfrak{M} = (W, Fop, D)$, em que

- $|W| \geq 2$;
- $Fop = \{f_{O_1}, ..., f_{O_n}\}$ é o conjunto de *funções de interpretação*, ou seja, as funções correspondentes a cada operador em $O = \{O_1, ..., O_n\}$ tais que $f_{O_i} : W^{n_{O_i}} \longrightarrow W$, n_{O_i} é a aridade de O_i;
- $D \subseteq W$, $D \neq \emptyset$ é o conjunto dos *valores designados*.

1.1.14. DEFINIÇÃO. Uma *linguagem de primeira ordem* ou *de predicados* $\mathcal{L}^{Pred}$ é a linguagem $\mathcal{L}^{Prop}$ aumentada com os seguintes conjuntos enumeráveis:

- $Q = \{Q_1, ..., Q_n\}$ de quantificadores;
- $V = \{x, y, z, x_1, y_1, z_1, ...\}$ de *variáveis nominais/individuais;*
- $Cons = \{a, b, c, a_1, b_1, c_1, ...\}$ de *constantes individuais*;
- $Pred = \{P, Q, R, P_1, Q_1, R_1, ...\}$ de predicados;
- $Fun = \{f, g, h, f_1, g_1, h_1, ...\}$ de funções.

1.1.15. DEFINIÇÃO. O conjunto F^* de $\mathcal{L}^{Pred}$ é definido como F em $\mathcal{L}^{Prop}$; as fórmulas atómicas são de igual modo tratadas como variáveis proposicionais e a seguinte condição adicional é satisfeita:

- se ϕ pertence a F^* e x é uma variável, então $Q_i x\,(\phi) \in F^*$.

1.1.16. DEFINIÇÃO. Na expressão $Q_i x\,(\phi)$, (ϕ) é o *escopo* do quantificador Q_i. Diz-se de toda a ocorrência de uma variável x no escopo de um quantificador Q_i que é *ligada*; diz-se de uma variável não ligada numa fórmula que é *livre* nessa fórmula.

1.1.17. DEFINIÇÃO. Diz-se que uma fórmula $\phi \in F^*$ está na *forma normal prenex* (FNP) se e só se (sse) ϕ tem a forma

$$Q_1 x_1, ..., Q_n x_n(M)$$

em que todo $Q_i x_i$, $i = 1, ..., n$ é quer $\forall x_i$ quer $\exists x_i$ e M é uma fórmula sem quantificadores. $Q_1 x_1, ..., Q_n x_n$ é o *prefixo* e (M) é a *matriz* de ϕ (não confundir com a matriz lógica $\mathfrak{M}$; cf. def. 1.1.13).

Toda a fórmula $\phi \in F^*$ pode ser transformada numa fórmula em FNP de modo algorítmico (ver abaixo; ver ainda, por exemplo, Chang & Lee, 1973, pp. 37-8).

1.1.4 A lógica clássica

A lógica clássica (LC) é fundamental no sentido em que as muitas e diversas lógicas contemporâneas são quer extensões quer generalizações dela, comummente diferindo

dela com respeito à noção central de consequência. Introduzimos agora alguns aspetos elementares de LC. (Um tratamento mais detalhado pode ser encontrado em, por exemplo, Mendelson (2009).)

1.1.18. DEFINIÇÃO (lógica clássica). Se dados $\mathcal{L}^{Prop}$ e/ou $\mathcal{L}^{Pred}$ nos restringimos ao conjunto de valores de verdade W tal que $W_2 = \{\mathrm{V(erdadeiro)}, \mathrm{F(also)}\}$, e definimos termos, átomos e fórmulas do modo seguinte

$$\begin{array}{lrcl} \text{termos} & t & := & x \mid f\,(t_1, ..., t_n) \\ \text{átomos} & P\ (Q, ...) & := & P\,(t_1, ..., t_n) \\ \text{fórmulas} & A\ (B, ...) & := & P \mid \neg A \mid A \wedge B \mid A \vee B \mid A \to B \mid \\ & & & A \leftrightarrow B \mid\ (\forall x)\, A \mid\ (\exists x)\, A \end{array}$$

então temos a teoria dos conetores lógicos $\neg$ (*negação*), $\wedge$ (*conjunção*), $\vee$ (*disjunção*), $\to$ (*implicação material*), e $\leftrightarrow$ (*equivalência material*) e dos quantificadores $\forall$ ("para todo") e $\exists$ ("existe um" or "para algum") conhecida como lógica clássica.

Referir-nos-emos à linguagem proposicional de LC como *lógica proposicional clássica* (LPC) e à linguagem de primeira ordem de LC como *lógica de primeira ordem clássica* (LPOC). A interpretação para LPC mais difundida é a das tabelas de verdade, nomeadamente das tabelas de verdade para $W_2 = \{\mathrm{V}, \mathrm{F}\}$ (ou $\{1, 0\}$, correspondentemente):

A	$\neg A$
V	F
F	V

A	B	$A \wedge B$	$A \vee B$	$A \to B$	$A \leftrightarrow B$
V	V	V	V	V	V
V	F	F	V	F	F
F	V	F	V	V	F
F	F	F	F	V	V

1.1.19. DEFINIÇÃO (equivalência lógica). Diz-se de duas fórmulas A e B que são *(logicamente) equivalentes* (denotado por $A \equiv B$) sse ambas recebem o mesmo valor de verdade em todas as atribuições de valores de verdade, ou seja, sse as suas tabelas de verdade para o conetor principal são idênticas.

1.1.20. DEFINIÇÃO. Uma fórmula de LPC é uma *tautologia* se toma uniformemente o valor de verdade V em toda e qualquer atribuição de valores de verdade às suas variáveis proposicionais. Denota-se uma tautologia arbitrária pelos símbolos $\top$ ou $\blacksquare$.

Temos assim o conjunto $TAUT$ de todas as tautologias de LPC:

$$TAUT = \{\phi \in F \mid val\,(\phi) = \text{V para toda } val : F \longrightarrow W_2\}\,.$$

1.1.21. RESULTADO. As fórmulas seguintes são tautologias bem conhecidas de LPC:

(T1)	$A \vee \neg A$	(lei do terceiro excluído)
(T2)	$\neg(A \wedge \neg A)$	(princípio de não contradição)
(T3)	$A \to A$	(lei da identidade)
(T4)	$\neg\neg A \leftrightarrow A$	(lei da negação dupla)
(T5)	$((A \to B) \wedge A) \to B$	(*modus ponens*)
(T6)	$((A \to B) \wedge \neg B) \to \neg A$	(*modus tollens*)
(T7)	$A \vee \top$	

Demonstração. A demonstração é por examinação das tabelas de verdade. □

1.1.22. DEFINIÇÃO (contrariedade). Diz-se de uma fórmula de LPC que é uma *contradição* se ela toma uniformemente o valor de verdade F em toda e qualquer atribuição de valores de verdade às suas variáveis proposicionais, ou seja,

$$val\,(\phi) = \text{F para toda } val : F \longrightarrow W_2.$$

1.1.23. PROPOSIÇÃO. Denota-se uma contradição arbitrária pelos símbolos $\perp$ ou $\square$. Temos imediatamente que

$$(*) \qquad A \wedge \neg A = \square$$

bem como

$$(**) \qquad A \wedge \square = \square.$$

Obviamente, a negação de uma tautologia é uma contradição, tal como o são as fórmulas $A \leftrightarrow \neg A$, $(A \wedge B) \wedge \neg (A \vee B)$ e $(\neg A \vee \neg B) \leftrightarrow (A \wedge B)$, por exemplo.

1.1.5 Formas normais e lógica clausal

Estamos agora em posição de apresentar um algoritmo para as FNPs. Dada uma fórmula ϕ em LPO, podemos transformá-la numa FNP em apenas quatro passos:

1. Remover todos os conetores $\rightarrow$ e $\leftrightarrow$ de acordo com as seguintes definições:

$$(\rightarrow_{def}) \qquad A \rightarrow B \quad := \quad \neg A \vee B$$

$$(\leftrightarrow_{def}) \qquad A \leftrightarrow B \quad := \quad (A \rightarrow B) \wedge (B \rightarrow A)$$

2. "Empurrar para dentro" os conetores $\neg$ usando as leis de de Morgan (DM1) e (DM2) e as regras (Q1) e (Q2):

$$\text{(DM1)} \qquad \neg (A \wedge B) \equiv \neg A \vee \neg B$$

$$\text{(DM2)} \qquad \neg (A \vee B) \equiv \neg A \wedge \neg B$$

$$\text{(Q1)} \qquad \neg \forall x \, (A) \equiv \exists x \, (\neg A)$$

$$\text{(Q2)} \qquad \neg \exists x \, (A) \equiv \forall x \, (\neg A)$$

3. Renomear as variáveis ligadas de modo que cada uma delas ocorra apenas uma vez.

4. "Empurrar para fora" os quantificadores utilizando as equivalências seguintes (desde que x não ocorra em A):

$$A \wedge \forall x \, (B) \equiv \forall x \, (A \wedge B)$$

$$A \vee \forall x \, (B) \equiv \forall x \, (A \vee B)$$

$$A \wedge \exists x \, (B) \equiv \exists x \, (A \wedge B)$$

$$A \vee \exists x \, (B) \equiv \exists x \, (A \vee B)$$

A FNP é apenas uma de algumas *formas normais.* A obtenção de formas normais é essencial para muitos procedimentos de demonstração. Aqui deixamos as definições

básicas da *lógica clausal*, baseada em formas normais conjuntivas e nas respetivas *cláusulas* (def. 1.1.31 abaixo).

1.1.24. DEFINIÇÃO. Definimos um *literal* como sendo um átomo (P) ou a negação de um átomo ($\neg P$). Dizemos que os literais P e $\neg P$ são *complementares*.

Referir-nos-emos frequentemente a um literal arbitrário como L e à sua negação como $\neg L$.

1.1.25. DEFINIÇÃO. Diz-se de uma fórmula A que está numa *forma normal conjuntiva* (FNC) sse A tem a forma $A = A_1 \wedge ... \wedge A_n$, $n \geq 1$, em que cada $A_1, ..., A_n$ é uma disjunção de literais, ou seja,

$$A = \bigwedge_{i=1}^{n} \left(\bigvee_{j=1}^{m} L_{i,j} \right).$$

Sejam $A_1, ..., A_n$ fórmulas. Abrevia-se $A_1 \wedge ... \wedge A_n$ como

$$\bigwedge_{i=1}^{n} A_i$$

e $A_1 \vee ... \vee A_n$ como

$$\bigvee_{i=1}^{n} A_i.$$

1.1.26. DEFINIÇÃO. Diz-se de uma fórmula A que está numa *forma normal disjuntiva* (FND) sse A tem a forma $A = A_1 \vee ... \vee A_n$, $n \geq 1$, em que cada $A_1, ..., A_n$ é uma conjunção de literais, ou seja,

$$A = \bigvee_{i=1}^{n} \left(\bigwedge_{j=1}^{m} L_{i,j} \right).$$

Mais rigorosamente, tem-se que uma fórmula em FNC tem a forma $\bigwedge_{i=1}^{n} \left(\bigvee_{j=1}^{m_i} L_{i,j} \right)$ e uma fórmula em FND tem a forma $\bigvee_{i=1}^{n} \left(\bigwedge_{j=1}^{m_i} L_{i,j} \right)$.

1.1.27. PROPOSIÇÃO. Seja $\{A_1, ..., A_n\}$ um conjunto finito de fórmulas. Então,

$$\neg\left(\bigwedge_{i=1}^{n} A_i\right) \equiv \left(\bigvee_{i=1}^{n} \neg A_i\right)$$

e

$$\neg\left(\bigvee_{i=1}^{n} A_i\right) \equiv \left(\bigwedge_{i=1}^{n} \neg A_i\right).$$

Demonstração. (Esboço) Temos que $\neg(A) \equiv (\neg A)$. Logo, obviamente a proposição está demonstrada para $n = 1$ por $\neg\left(\bigwedge_{i=1}^{1} A_i\right) \equiv \left(\bigvee_{i=1}^{1} \neg A_i\right)$. A demonstração segue então por indução sobre n. □

1.1.28. PROPOSIÇÃO. Sejam A uma fórmula em FNC e B uma fórmula em FND. Então $\neg A$ é equivalente a uma fórmula em FND e $\neg B$ é equivalente a uma fórmula em FNC.

Demonstração. Se A está em FNC, então A é a fórmula $\bigwedge_{i=1}^{n}\left(\bigvee_{j=1}^{n} L_{i,j}\right)$. Pela proposição 1.1.26, $\neg A = \neg\bigwedge_{i=1}^{n}\left(\bigvee_{j=1}^{n} L_{i,j}\right) \equiv \bigvee_{i=1}^{n}\neg\left(\bigvee_{j=1}^{n} L_{i,j}\right) \equiv \bigvee_{i=1}^{n}\left(\bigwedge_{j=1}^{n}\neg L_{i,j}\right)$. A demonstração segue de modo semelhante para $\neg B$ equivalente a uma fórmula em FNC. □

Toda e qualquer fórmula pode ser transformada numa FNC ou numa FND em três passos principais aplicando as definições ($\rightarrow_{def}$) e ($\leftrightarrow_{def}$), juntamente com as leis da eliminação da dupla negação (T4), de de Morgan (DM1-2) e da distribuição. (Cf., por exemplo, Chang & Lee, 1973, p. 14.)

1.1.29. TEOREMA. Toda a fórmula A é equivalente a uma fórmula A_1 em FNC e a uma fórmula A_2 em FND.

Demonstração. Segue imediatamente do material acima. A demonstração é por indução sobre a complexidade de A. □

O teorema 1.1.29 garante a existência de uma fórmula em FND equivalente a uma fórmula A. De modo a encontrar essa fórmula basta computar a tabela de verdade de A. Considerando-se as linhas em que A é verdadeira obtemos a fórmula em FND equivalente a A.

1.1.30. EXEMPLO. Seja que $A = (B \vee C) \wedge ((\neg B \wedge C) \vee D)$. A tabela de verdade de A apresenta-se do seguinte modo:

B	C	D	A	$\neg A$
V	V	V	V	F
V	V	F	F	V
V	F	V	V	F
V	F	F	F	V
F	V	V	V	F
F	V	F	V	F
F	F	V	F	V
F	F	F	F	V

Temos então que a FND de A é $(B \wedge C \wedge D) \vee (B \wedge \neg C \wedge D) \vee (\neg B \wedge C \wedge D) \vee (\neg B \wedge C \wedge \neg D)$. De igual modo, considerando-se negativamente as linhas em que A é falsa, i.e., considerando-se a FND de $\neg A$ e subsequentemente negando-a obtém-se a fórmula em FNC equivalente a A. Assim sendo, a FND de $\neg A$ é $(B \wedge C \wedge \neg D) \vee (B \wedge \neg C \wedge \neg D) \vee (\neg B \wedge \neg C \wedge D) \vee (\neg B \wedge \neg C \wedge \neg D)$; pela proposição 1.1.27, a negação desta FND produz a FNC $(\neg B \vee \neg C \vee D) \wedge (\neg B \vee C \vee D) \wedge (B \vee C \vee \neg D) \wedge (B \vee C \vee D)$.

O método apresentado acima de transformação de uma fórmula numa FNC *equivalente* pode resultar numa "explosão" exponencial da fórmula. Pode-se evitar uma tal explosão com métodos que preservam apenas a *satisfazibilidade* da fórmula original, mas que garantem um aumento linear. Um exemplo de um tal método é o de Tseitin (1968) (ver por exemplo Harrison, 2009, p. 73, para uma apresentação sucinta deste método).

1.1.31. DEFINIÇÃO. Uma *cláusula* é uma disjunção finita de literais. Uma cláusula com um literal apenas é uma *cláusula unitária*. A *cláusula vazia* é a cláusula que não contém nenhum literal.

Referimo-nos a uma cláusula arbitrária como C e denotamos a cláusula vazia pelo símbolo $\square$. Designaremos um conjunto de cláusulas por $\mathcal{C}$.

1.1.32. PROPOSIÇÃO. A cláusula vazia é sempre falsa.

Demonstração. A falsidade da cláusula vazia deriva do facto que ela não tem nenhum literal que possa ser satisfeito por uma interpretação. □

1.1.33. DEFINIÇÃO. Uma expressão ou uma cláusula diz-se *básica* (*ground*) se nela não ocorre qualquer variável.

1.1.34. DEFINIÇÃO. C é uma *cláusula de Horn* se contém no máximo um literal positivo. Uma cláusula de Horn com exatamente um literal positivo é uma *cláusula definida*. C é uma *cláusula de Horn dual* se tem no máximo um literal negativo.

1.1.6 Fórmulas assinaladas e lógica assinalada (introdução)

1.1.35. DEFINIÇÃO. Uma *fórmula assinalada* tem a forma $\blacktriangle [A_j]$ para $1 \leq j \leq n$ e em que $\blacktriangle$ é o *sinal* de A_j. O sinal de uma fórmula pode ser um valor de verdade isolado $v_i \in W$, caso que se denota por $v_i [A]$, ou um conjunto de valores de verdade $S \subseteq W$, denotando-se este último caso por $S [A]$.

Este formalismo, conhecido por *lógica assinalada*, tem um papel importante na automatização da demonstração de teoremas em lógicas multivalentes, e será devidamente desenvolvido no Capítulo 4.

1.2 A consequência lógica

Como se disse acima, a consequência lógica é uma noção central em qualquer sistema lógico. Passamos a abordar os vários aspetos que estão diretamente dependentes desta noção central.

1.2.1 A consequência tarskiana

Do ponto de vista da operação de consequência Cq, um sistema lógico $\mathrm{S} = (\mathcal{L}, Cq)$ é a teoria de que fórmulas (mais geralmente: asserções ou frases) se seguem necessariamente (ou seja, são deriváveis) de que outras fórmulas.

1.2.1. DEFINIÇÃO. Para um sistema lógico $\mathrm{S} = (\mathcal{L}, Cq)$, uma *operação tarskiana de consequência* Cq é um mapeamento $Cq : 2^F \longrightarrow 2^F$ que satisfaz as seguintes condições:

Para todo $X, Y \subseteq F$,

$(Cq1)$	$X \subseteq Cq(X)$	(inclusão)
$(Cq2)$	$Cq(Cq(X)) \subseteq Cq(X)$	(idempotência)
$(Cq3)$	$Cq(X) \subseteq Cq(Y)$ sempre que $X \subseteq Y$	(monotonocidade)

Para todo $X \subseteq F$, $F \subseteq \mathcal{L}$, $Cq(X)$ é o conjunto de todas as consequências de X e $\phi \in Cq(X)$ denota o facto que ϕ é uma consequência de X em $\mathcal{L}$, ou, o que é o mesmo, que ϕ pode ser inferido de X em $\mathcal{L}$.

Se adicionalmente a $(Cq1) - (Cq3)$ para qualquer substituição $e \in End(\mathcal{L})$ temos que

$$\text{(Sub)} \qquad eCq(X) \subseteq Cq(eX)$$

dizemos que Cq é *estrutural*.

Um **morfismo** é um mapeamento de um objeto matemático (o *domínio*) para outro (o *contradomínio*) que preserva a estrutura dos objetos. Seja f um morfismo com domínio X e contradomínio Y; escreve-se então $f : X \longrightarrow Y$. Se X e Y são conjuntos, então f é uma função.
Um **endomorfismo** é um morfismo $f : X \longrightarrow X$ (denotado por $End(X)$).

1.2.2. Definição. Equivalentemente, pode-se considerar uma *relação tarskiana de consequência* $R \subseteq 2^F \times F$ para um par $(\mathcal{L}, R)$ satisfazendo as propriedades seguintes em que $(X, \phi) \in R$ denota que ϕ é uma consequência de X:

1. Se $\phi \in R$, então $(X, \phi) \in R$.
2. Se $(X, \phi) \in R$ e $(Y, \psi) \in R$ para todo $\psi \in X$, então $(Y, \phi) \in R$.
3. Se $(X, \phi) \in R$ e $X \subseteq Y$, então $(Y, \phi) \in R$.

Dado que a relação de consequência R é vista como uma relação entre um conjunto de fórmulas e uma única fórmula, toda a relação de consequência R define uma operação de consequência Cq e vice-versa.

1.2.3 PROPOSIÇÃO. Pode mostrar-se que toda a consequência de matriz $Cn_{\mathfrak{M}}$ (ver abaixo) é estrutural e, conversamente, toda a consequência estrutural Cq de $\mathcal{L}$ e todo o conjunto de fórmulas X determinam uma matriz $\mathcal{L}_X = (\mathcal{L}, Cq(X))$ chamada a *matriz de Lindenbaum* para Cq; uma classe de matrizes envolvendo a mesma álgebra $\mathbf{L}_{Cq} = \{\mathcal{L}_X \mid X \subseteq \mathcal{L}\}$ chama-se um *"bundle" de Lindenbaum* para Cq.

Uma **álgebra** (abreviando **álgebra abstrata**) é um par

$$\mathcal{A} = (U, O)$$

em que $U \neq \emptyset$ é o universo de $\mathcal{A}$ e $O = \{f_1, ..., f_n\}$ é o conjunto de operações finitárias f_i em ou sobre U. Exemplos de álgebras são grupos (ex.: $(\mathbb{Z}, +)$), anéis (ex.: $(\mathbb{Z}, +, \cdot)$), campos (ex.: $(\mathbb{Q}, +, \cdot, -, 0, 1)$) e reticulados (ex.: $\mathbb{Z}^+, \div$), em que $\mathbb{Z}$ denota o conjunto de números inteiros $\{..., -2, -1, 0, 1, 2, ...\}$, $\mathbb{Z}^+$ denota o conjunto de números inteiros positivos e $\mathbb{Q}$ denota o conjunto de números racionais.

1.2.2 Dedução, validade e satisfazibilidade

Para um sistema lógico S, a operação de consequência Cq especifica a classe de inferências válidas em S, ou, noutros termos, que inferências em S preservam a verdade ou são dedutivamente válidas. Diz-se que uma inferência é dedutiva (vs. indutiva ou abdutiva) se a conclusão se segue necessariamente de um conjunto de premissas. A operação de consequência pode ser definida de um ponto de vista quer semântico quer sintático.

1.2.4. DEFINIÇÃO (validade). Seja Γ um conjunto de fórmulas e A uma fórmula derivada de Γ. Diz-se que A é *válida* sse não há nenhuma interpretação que atribua os valores de verdade V a todos os membros de Γ (as *premissas*) e F a A (a *conclusão*), e escreve-se $\Gamma \models A$. Uma fórmula é *inválida* sse não é válida.

1.2.5. DEFINIÇÃO (satisfazibilidade). (i) Diz-se que uma interpretação $\mathcal{I}$ *satisfaz* uma fórmula A sse $val_{\mathcal{I}}(A) = \text{V}$ e denota-se esta relação por $\mathcal{I} \models A$. Diz-se então que $\mathcal{I}$ é um *modelo* de A. Diz-se que uma fórmula A é *satisfazível* (ou *consistente*) sse existe uma interpretação que é um modelo de A. Caso contrário, diz-se que A é *insatisfazível* (ou *inconsistente*).

(ii) Diz-se que uma interpretação $\mathcal{I}$ *satisfaz* um conjunto de fórmulas $\Gamma = \{B_1, ..., B_n\}$

sse para todo $B_i \in \Gamma$, $\mathcal{I} \models B_i$. Logo, temos que um conjunto de fórmulas é *satisfazível* (ou *consistente*) sse todos os seus membros têm um modelo em comum. Tal implica que se pode ver Γ como uma conjunção das fórmulas B_i, i.e., $\Gamma \equiv \bigwedge_{i=1}^{n} B_i$. Caso contrário, diz-se que Γ é *insatisfazível* (ou *inconsistente*).

1.2.6. EXEMPLO. Seja $\mathcal{I}$ uma interpretação para $\mathcal{L}_{\Upsilon}$, Υ é uma assinatura (i.e., para uma fórmula $\phi \in \mathcal{L}_{,\Upsilon} := Pred(\phi) \cup Fun(\phi) \cup Cons(\phi)$) e $W = \{v_0, ..., v_{n-1}\}$. Temos que:

- $\mathcal{I} \models \top$ e $\mathcal{I} \not\models \bot$
- $\mathcal{I} \models \phi \wedge \psi$ sse $\mathcal{I} \models \phi$ e $\mathcal{I} \models \psi$
- $\mathcal{I} \models \phi \vee \psi$ sse $\mathcal{I} \models \phi$ ou $\mathcal{I} \models \psi$
- $\mathcal{I} \models \phi \rightarrow \psi$ sse $\mathcal{I} \not\models \phi$ ou $\mathcal{I} \models \psi$
- $\mathcal{I} \models \phi \leftrightarrow \psi$ sse $\mathcal{I} \models \phi$ e $\mathcal{I} \models \psi$, ou $\mathcal{I} \not\models \phi$ e $\mathcal{I} \not\models \psi$
- $\mathcal{I} \models \neg\phi$ sse $\mathcal{I} \not\models \phi$
- $\mathcal{I} \models (\forall x)\, \phi$ sse $\mathcal{I}_d^x \models \phi$ para todo $d \in \mathscr{D}$
- $\mathcal{I} \models (\exists x)\, \phi$ sse $\mathcal{I}_d^x \models \phi$ para algum $d \in \mathscr{D}$

1.2.7. DEFINIÇÃO. Uma fórmula A é uma *consequência lógica* de um conjunto de fórmulas Γ sse toda a interpretação que satisfaz Γ também satisfaz A. Denota-se esta relação por $\Gamma \models A$.

É agora óbvio que uma fórmula A é válida sse (1) toda a interpretação de A é também um modelo de A e (2) A é uma tautologia. É ainda óbvio que

- Uma fórmula é válida sse a sua negação é insatisfazível.
- Uma fórmula é insatisfazível sse a sua negação é válida.
- Uma fórmula é inválida sse existe pelo menos uma interpretação que a falsifica.
- Uma fórmula é satisfazível sse existe pelo menos uma interpretação que a torna verdadeira.
- Se uma fórmula é válida, então é satisfazível (mas o contrário não se verifica).

- Se uma fórmula é insatisfazível, então é inválida (mas o contrário não se verifica).

1.2.8. TEOREMA (teorema da dedução). $\Gamma \models A$ sse $\Gamma \cup \{\neg A\}$ é insatisfazível.

Demonstração. ($\Rightarrow$) Para qualquer interpretação $\mathcal{I}$ verifica-se que quer $val_{\mathcal{I}}(B) = \mathrm{V}$ para todo $B \in \Gamma$ e $val_{\mathcal{I}}(A) = \mathrm{V}$ (logo, $val_{\mathcal{I}}(\neg A) = \mathrm{F}$), quer $val_{\mathcal{I}}(B) = \mathrm{F}$ para algum $B \in \Gamma$. De qualquer modo, temos que $val_{\mathcal{I}}(\Gamma \cup \{\neg A\}) = \mathrm{F}$.

($\Leftarrow$) Para qualquer interpretação $\mathcal{I}$, temos que ou $val_{\mathcal{I}}(B) = \mathrm{V}$ para todo $B \in \Gamma$ e $val_{\mathcal{I}}(\neg A) = \mathrm{F}$ (logo, $val_{\mathcal{I}}(A) = \mathrm{V}$), ou $val_{\mathcal{I}}(B) = \mathrm{F}$ para algum $B \in \Gamma$. Logo, $val_{\mathcal{I}}(A) = \mathrm{V}$ sempre que $val_{\mathcal{I}}(\Gamma) = \mathrm{V}$ e temos então que $\Gamma \models A$. □

Será útil dar agora uma formulação equivalente do teorema da dedução:

1.2.9. TEOREMA (teorema da dedução). Dado um conjunto de fórmulas $\Gamma = \{B_1, ..., B_n\}$ e uma fórmula A, A é uma consequência lógica de Γ sse a fórmula $((B_1 \wedge ... \wedge B_n) \rightarrow A)$ é válida.

Demonstração. A prova segue-se imediatamente do material acima. □

O que se disse acima permite-nos agora uma reformulação da definição de equivalência lógica:

1.2.10. DEFINIÇÃO. Duas fórmulas A e B são *equivalentes* sse (1) têm exatamente o mesmo valor de verdade em todos os modelos, (2) $A \models B$ e $B \models A$, e (3) $A \leftrightarrow B$ é uma tautologia.

1.2.11. TEOREMA. Uma fórmula A é uma *consequência lógica* de um conjunto de fórmulas $\Gamma = \{B_1, ..., B_n\}$ sse a fórmula $(B_1 \wedge ... \wedge B_n \wedge \neg A)$ é insatisfazível.

Demonstração. Segue-se do teorema 1.2.9 que A é uma consequência lógica de $\Gamma = \{B_1, ..., B_n\}$ sse a negação de $((B_1 \wedge ... \wedge B_n) \rightarrow A)$ é insatisfazível. Com efeito, temos que $\neg((B_1 \wedge ... \wedge B_n) \rightarrow A) \equiv B_1 \wedge ... \wedge B_n \wedge \neg A$ (por aplicação da definição de $\rightarrow$, das leis de de Morgan e da propriedade de associatividade). □

Mas a consequência lógica pode estabelecer-se antes de, e mesmo sem, uma interpretação através da aplicação de regras puramente sintáticas, caso em que se aplicam as seguintes definições:

1.2.12. DEFINIÇÃO. Diz-se que Γ *prova* A, ou que A é *demonstrável* or *dedutível* de Γ, e escreve-se $\Gamma \vdash A$, se A se obtém de Γ por meio da aplicação finita de uma ou mais regras de inferência.

1.2.13. DEFINIÇÃO. Uma *regra de inferência* é uma expressão com a forma geral

$$\frac{\Gamma_1 \vdash A_1, ..., \Gamma_n \vdash A_n}{\Gamma \vdash A}$$

em que $\Gamma \vdash A$ é um *sequente*, uma expressão composta por um *antecedente* Γ e um *sucedente* A, ambos sequências de fórmulas.

Dado um sistema lógico $\mathrm{S} = (\mathcal{L}, \vdash)$, se $\emptyset \vdash A$ (i.e., $A \in Cq(\emptyset)$), escrevemos então simplesmente $\vdash A$.

1.2.14. DEFINIÇÃO. A *lógica* de S, denotada por Λ_{S}, é o conjunto dos *teoremas* ($\Lambda_{\mathrm{S}} = \{\phi \in \mathcal{L} \mid \vdash \phi\}$) ou *fórmulas válidas* ($\Lambda_{\mathrm{S}} = \{\phi \in \mathcal{L} \mid \models \phi\}$) de S.

1.2.3 Correção e completude

Ambas as noções – semântica e sintática – de consequência lógica são fulcrais para um sistema lógico na medida em que exprimem os resultados metateóricos fundamentais de *correção* e *completude*. Seja S um sistema lógico com uma teoria semântica.[2]

1.2.15. DEFINIÇÃO (correção). S é *correto* se, se $\Gamma \vdash_S \phi$, então $\Gamma \models_S \phi$.

1.2.16. DEFINIÇÃO (completude). S é *completo* se, se $\Gamma \models_S \phi$, então $\Gamma \vdash_S \phi$.

A definição 1.2.15 significa que tudo o que é demonstrável em S é de igual modo logicamente verdadeiro, ou seja, S não demonstra falsidades. Quanto à definição 1.2.16, esta significa que todas as verdades lógicas em S são demonstráveis em S.

[2]Ou, o que é o mesmo, sejam $\mathrm{S} = (\mathcal{L}, \vdash)$ e $\mathrm{T} = (\mathcal{L}, \models)$ sistemas lógicos sintática e semanticamente derivados, respetivamente, numa mesma linguagem $\mathcal{L}$. Então, avalia-se a correção e a completude de S com respeito a T.

Juntas, a correção e a completude exprimem o facto que num sistema lógico se pode derivar tudo o que deve ser derivado (o sistema é completo) e nada que não deva ser derivado (o sistema é correto). É óbvio por estas definições que S é correto se $\vdash_S \subseteq \models_S$ e *fortemente* completo se $\models_S \subseteq \vdash_S$.

1.2.4 Sistemas axiomáticos

É prática comum especificar uma operação/relação de consequência por meio de um *sistema axiomático* (ou *dedutivo*), i.e., um conjunto de *esquemas de axiomas* juntamente com um conjunto de regras de inferência. Com efeito, as lógicas estruturais caracterizam-se por meio dos seus axiomas e regras de inferência. A lógica Λ derivada a partir de um tal sistema é o conjunto de todos os axiomas e de todas as fórmulas fechado sob as regras de inferência.

1.2.17. Exemplo (LPCA). O sistema axiomático para a lógica proposicional clássica conhecido por sistema de Frege-Łukasiewicz e que designamos como LPCA é constituído por:

- Três esquemas de axiomas:

$$\begin{array}{ll} \text{(LPC1)} & A \to (B \to A) \\ \text{(LPC2)} & (A \to (B \to C)) \to ((A \to B) \to (A \to C)) \\ \text{(LPC3)} & (\neg A \to \neg B) \to (B \to A) \end{array}$$

- Uma regra de inferência (*modus ponens*):

$$\text{(MP)} \qquad A, A \to B \vdash B$$

Com se pode ver, os conetores $\neg$ e $\to$ são primitivos, e os restantes três conetores definem-se do modo que se segue:

$$\begin{array}{lcl} A \vee B & := & \neg A \rightarrow B \\ A \wedge B & := & \neg(A \rightarrow \neg B) \\ A \leftrightarrow B & := & (A \rightarrow B) \wedge (B \rightarrow A), \\ & & \neg((A \rightarrow B) \rightarrow \neg(B \rightarrow A)) \end{array}$$

LPCA é correto e completo, sendo logo adequado para LPC de acordo com a definição seguinte:

1.2.18. DEFINIÇÃO (adequação). Diz-se que um sistema axiomático X é *adequado* para um sistema lógico S sempre que o conjunto de tautologias derivadas por meio dos seus conetores primitivos coincide com o conjunto dos teoremas de S, ou seja, quando a axiomatização de S por meio de X é igualmente correta e completa.

1.3 Resultados elementares adicionais

1.3.1 Matrizes e álgebras lógicas

Concentramo-nos agora em $\mathcal{L}$ de um ponto de vista algébrico. É com frequência produtivo (ex.: Rasiowa, 1974; Stachniak, 1996) ver uma linguagem $\mathcal{L}^{Prop} = (F, O_1, ..., O_m)$ como uma *álgebra de fórmulas* em que F é um conjunto de fórmulas, $O_1, ..., O_m$ são operações finitárias sobre F e a álgebra $\mathcal{L}^{Prop}$ é gerada livremente (ver abaixo) pelo conjunto Vp de variáveis proposicionais. No caso específico de LPC temos que

$$\mathcal{L}_k^{Prop} = (F, \neg, \rightarrow, \vee, \wedge, \leftrightarrow)$$

é uma álgebra abstrata de tipo $(1, 2, 2, 2, 2)$.

Diz-se de uma álgebra que é de **tipo** $(n_1, ..., n_k)$ para n_i a aridade das suas operações. Por exemplo, um grupo é uma álgebra de tipo (2).

1.3.1. PROPOSIÇÃO (Axioma de Frege). Seja ϕ uma fórmula de $\mathcal{L}^{Prop}$; de modo a interpretar $\mathcal{L}^{Prop}$ temos de atribuir a ϕ um *significado*. O significado de ϕ é o seu *correlato semântico*. Seja $\mathscr{A}$ o contradomínio de todos os correlatos semânticos; duas condições têm de ser satisfeitas para um mapeamento $r : F \rightarrow \mathscr{A}$ (Frege, 1892):

- (*) Há exatamente um correlato semântico associado a cada $\phi \in F$ (i.e., r é uma função).
- (**) (Princípio de extensionalidade) Duas fórmulas $\xi, \psi \in F$ podem substituir-se mutuamente num qualquer contexto proposicional $\phi \in F$ sempre que $r(\xi) = r(\psi)$ ou, noutras palavras, quando para cada $\phi \in F$, $p \in Vp$ temos que

$$r(\phi(\xi/p)) = r(\phi(\psi/p)) \text{ sse } r(\xi) = r(\psi),$$

em que $\phi(\xi/p)$ e $\phi(\psi/p)$ denotam as fórmulas que resultam de ϕ após a substituição $\xi(\psi)$ em vez de p.

De facto, (**) asserta que a denotação de uma proposição é uma função da denotação dos seus componentes. A versão puramente sintática do princípio de extensionalidade tem um papel importante em muitos sistemas axiomáticos, pelo que é pertinente introduzi-la aqui.

1.3.2. DEFINIÇÃO. Uma *substituição* de uma fórmula $\phi(p_1, ..., p_n)$ com no máximo $p_1, ..., p_n$ variáveis é uma qualquer fórmula $e\phi = \phi(ep_1/p_1, ..., ep_n/p_n)$ obtida a partir de ϕ pela substituição simultânea de $p_1, ..., p_n$ por uma qualquer fórmula $ep_1, ..., ep_n$.

1.3.3. PROPOSIÇÃO. A regra de *substituição*

$$\text{(SUB)} \qquad \phi / e\phi$$

asserta que se uma fórmula ϕ é um teorema, então qualquer uma das suas instâncias de substituição (i.e., fórmulas extensionalmente equivalentes) também é um teorema.

Uma interpretação para $\mathcal{L}^{Prop}$ pode ser uma estrutura

$$\mathcal{A} = (\mathscr{A}, f_1, ..., f_m)$$

que é uma álgebra semelhante a $\mathcal{L}^{Prop}$ de acordo com o seguinte teorema:

1.3.4. TEOREMA (Suszko, 1957). Se $\mathscr{A}$ é o conjunto de todos os correlatos semânticos da linguagem $\mathcal{L}^{Prop} = (F, O_1, ..., O_m)$ e a função r se define como na

proposição 1.3.1, então para todo $i = 1, ..., m$ a fórmula

$$r\left(O_i\left(\xi_1, ..., \xi_n\right)\right) = f_i\left(r\left(\xi_1\right), ..., r\left(\xi_n\right)\right)$$

define unicamente uma função f_i em $\mathscr{A}$ com a mesma aridade de O_i.

Todo o mapeamento $s : Vp \longrightarrow \mathscr{A}$ pode ser unicamente estendido ao homomorfismo $h_s : \mathcal{L} \longrightarrow \mathcal{A}$ tal que $h_s \in Hom\left(\mathcal{L}^{Prop}, \mathcal{A}\right)$ e $\mathcal{L}^{Prop}$ é uma álgebra absolutamente livre gerada por Vp na sua classe de semelhança.

Um **homomorfismo** é um morfismo entre duas estruturas algébricas semelhantes. Sejam $\mathcal{A}$ e $\mathcal{B}$ duas álgebras semelhantes (ex.: grupos); um homomorfismo de $\mathcal{A} = (X, f_1, ..., f_m)$ para $\mathcal{B} = (Y, g_1, ..., g_m)$, denotado por $Hom\left(\mathcal{A}, \mathcal{B}\right)$, é um mapeamento $h : X \longrightarrow Y$, em que X e Y são os universos de $\mathcal{A}$ e $\mathcal{B}$ respetivamente e se tem que para todo $\xi_1, ..., \xi_n \in X$, $n \geq 0$ é a aridade de f_i,

$$h\left(f_i\left(\xi_1, ..., \xi_n\right)\right) = f_i\left(h\left(\xi_1\right), ..., h\left(\xi_n\right)\right).$$

Por outras palavras, um homomorfismo preserva todas as operações. Por exemplo, sejam G e H dois grupos com a operação $+$; então, $h : G \longrightarrow H$ é um homomorfismo sse para todo $g_1, g_2 \in G$ se tem que $h\left(+\left(g_1, g_2\right)\right) = +\left(h\left(g_1\right), h\left(g_2\right)\right)$, ou numa notação mais habitual, $h\left(g_1 + g_2\right) = h\left(g_1\right) + h\left(g_2\right)$.

Diz-se que uma álgebra $\mathcal{A} = (X, f_1, ..., f_m)$ é **absolutamente livre** se existe um conjunto $G \subseteq X$ de geradores de $\mathcal{A}$ tal que todo o mapeamento s de G para o universo Y de uma álgebra semelhante $\mathcal{B} = (Y, g_1, ..., g_m)$ se pode estender a um homomorfismo de $\mathcal{A}$ para $\mathcal{B}$. Diz-se que os elementos de G são *geradores livres* de $\mathcal{A}$ e diz-se de $\mathcal{A}$ que é *livremente gerada* por G.

Dado isto, podemos agora reformular a definição de matriz lógica $\mathfrak{M}$, uma estrutura de interpretação com um subconjunto distinto de elementos correspondendo a frases de um tipo específico como, por exemplo, proposições verdadeiras (cf. def. 1.1.13):

1.3.5. DEFINIÇÃO. Uma *matriz lógica* $\mathfrak{M}$ é um par $(\mathcal{A}, D)$ em que $\mathcal{A}$ é uma álgebra semelhante à linguagem proposicional $\mathcal{L}^{Prop}$ e $D \subseteq \mathscr{A}$ é um conjunto não vazio do universo de $\mathcal{A}$ com D o conjunto dos valores designados de $\mathfrak{M}$. Associado

a toda a matriz $\mathfrak{M}$ há um conjunto de fórmulas

$$E(\mathfrak{M}) = \left\{\phi \in F \mid h(\phi) \in D \text{ para todo } h \in Hom\left(\mathcal{L}^{Prop}, \mathcal{A}\right)\right\}$$

chamado o *conteúdo* de $\mathfrak{M}$ e para toda a matriz $\mathfrak{M}$ define-se a relação $\models_{\mathfrak{M}}$ para todo $X \subseteq F, \phi \in F$,

$$X \models_{\mathfrak{M}} \phi \text{ sse para todo } h \in Hom\left(\mathcal{L}^{Prop}, \mathcal{A}\right), h(\phi) \in D \text{ sempre que } h(X) \subseteq D.$$

1.3.6. DEFINIÇÃO. A toda a relação $\models_{\mathfrak{M}}$ pode associar-se unicamente uma operação $Cn_{\mathfrak{M}} : 2^F \longrightarrow 2^F$ tal que

$$\phi \in Cn_{\mathfrak{M}}(X) \quad \text{sse} \quad X \models_{\mathfrak{M}} \phi$$

Diz-se então que $Cn_{\mathfrak{M}}$ é uma *operação de consequência matricial* e a relação $\models_{\mathfrak{M}}$ é uma *relação de consequência matricial* $\models_{\mathfrak{M}} \subseteq 2^F \times F$ para $\mathfrak{M}$, sendo que $\models_{\mathfrak{M}} \subseteq 2^F \times F$ é uma generalização natural da relação de consequência clássica Cq (cf. def. 1.2.1).

1.3.7. EXEMPLO. Para $\mathcal{L}_2^{Prop}$, i.e., a linguagem proposicional da lógica clássica, tem-se que a *álgebra de 2 elementos da lógica clássica* tem a forma

$$\mathcal{A}_2 = (\{0,1\}, \neg, \rightarrow, \vee, \wedge, \leftrightarrow)$$

enquanto a *matriz clássica* tem a forma

$$\mathfrak{M}_2 = (\{0,1\}, \neg, \rightarrow, \vee, \wedge, \leftrightarrow, \{1\}) = (\mathcal{A}_2, \{1\})$$

e a *relação de consequência clássica* caracteriza-se do seguinte modo:

$$X \models_{\mathfrak{M}_2} \phi \text{ sse, para todo } h \in Hom\left(\mathcal{L}_k^{Prop}, \mathcal{A}_2\right), h(\phi) = 1 \text{ sempre que } h(X) \subseteq \{1\}.$$

O conteúdo de $\mathfrak{M}_2$ é pois o conjunto que consiste nas fórmulas que são consequências do conjunto vazio,

$$E(\mathfrak{M}_2) = \{\phi \mid \models_2 \phi\} = TAUT$$

Obviamente, $TAUT$ é o conjunto de todas as tautologias clássicas, i.e., fórmulas que são esquemas exclusivos de proposições verdadeiras de LC.

A propriedade distintiva da lógica gerada por $\mathfrak{M}$ é a sua *estruturalidade* (cf. def. 1.2.2). Com efeito, para toda a substituição $e \in End(\mathcal{L})$

$X \models_{\mathfrak{M}} \phi$ implica que $eX \models_{\mathfrak{M}} e\phi$, ou de modo equivalente,
$\phi \in Cn_{\mathfrak{M}}(X)$ implica que $e\phi \in Cn_{\mathfrak{M}}(eX)$.

A propriedade fundamental das lógicas estruturais é a sua representação por meio de uma matriz:

1.3.8. Definição (representação matricial). Para toda a operação de consequência Cq existe uma classe de matrizes $\mathbf{K}$ tal que para todo $X \in F$,

$$Cq(X) = \cap \{Cn_{\mathfrak{M}}(X) \mid \mathfrak{M} \in \mathbf{K}\}.$$

Isto pode reformular-se como o *teorema da completude* procedendo do seguinte modo:

1.3.9. Teorema (Malinowski, 1989). Chame-se a uma qualquer classe $\mathbf{K}$ de matrizes para uma linguagem $\mathcal{L}^{Prop}$ uma *semântica matricial* para $\mathcal{L}^{Prop}$; então, toda a semântica $\mathbf{K}$ para $\mathcal{L}^{Prop}$ define uma função $Cn_{\mathbf{K}} : \mathcal{L}^{Prop} \supseteq X \longrightarrow Cn_{\mathbf{K}}(X) \subseteq \mathcal{L}^{Prop}$ tal que para todo $X \subseteq \mathcal{L}^{Prop}$ e para $\phi \in \mathcal{L}^{Prop}$, $\phi \in Cn_{\mathbf{K}}(X)$ sse para toda a matriz $\mathfrak{M} = (\mathcal{A}, D) \in \mathbf{K}$ e toda a atribuição $h \in Hom\left(\mathcal{L}^{Prop}, \mathcal{A}\right)$, se $h(X) \subseteq D$, então $h(\phi) \in D$. Segue-se que se $\mathbf{K}$ é uma semântica para $\mathcal{L}^{Prop}$, então a função $Cn_{\mathbf{K}}$ é uma *operação de consequência estrutural* em $\mathcal{L}^{Prop}$, e diz-se que uma lógica $\left(\mathcal{L}^{Prop}, Cq\right)$ é *fortemente completa* em relação a $\mathbf{K}$ para $\mathcal{L}^{Prop}$ se $Cq = Cn_{\mathbf{K}}$.

1.3.10. Teorema (Lindenbaum; Wójcicki, 1969). Toda e qualquer lógica é fortemente completa com respeito a um *bundle* de Lindenbaum $\mathbf{L}_{Cq}$.

Demonstração. Pode encontrar-se uma demonstração em Malinowski (1989). □

Note-se que o teorema 1.3.10 implica que toda e qualquer lógica tem uma semântica verofuncional, nomeadamente uma semântica matricial.

1.3.2 Multivalência lógica

Falamos trivialmente de *multivalência lógica* quando $|W| > 2$. Estamos agora em posição de definir critérios não triviais para a multivalência lógica. Com efeito, o modo mais natural de obter multivalência lógica é por meio das matrizes lógicas. Uma vez escolhida uma linguagem proposicional (tipicamente, $\mathcal{L}_k^{Prop}$ ou um dos seus redutos), define-se uma álgebra de múltiplos elementos $\mathcal{A}$ semelhante a ela juntamente com o conjunto de valores designados $D \subseteq \mathscr{A}$. Dizemos que a matriz resultante $\mathfrak{M} = (\mathcal{A}, D)$ define uma lógica multivalente sempre que $E(\mathfrak{M})$ não pode ser descrito por uma matriz de dois elementos.

Seja $\mathcal{A} = (X, f_1, ..., f_m)$ uma álgebra abstrata. Diz-se que uma álgebra $\mathcal{B} = (X, f_1, ..., f_l)$ é um **reduto** de $\mathcal{A}$ se $m \in I$, $l \in K$ e $K \subsetneq I$.

O problema posto nestes termos, usamos dois critérios para decidir quando é que uma matriz $\mathfrak{M}$ determina uma lógica multivalente (Malinowski, 1993):

1.3.11. Proposição. Uma matriz lógica $\mathfrak{M}_{n>2}$ determina uma lógica multivalente sse *não* existe nenhuma matriz de dois elementos $\mathfrak{M}_2$ para $\mathcal{L}^{Prop}$ que satisfaça as seguintes igualdades:

1. $E(\mathfrak{M}_{n>2}) = E(\mathfrak{M}_2)$;
2. $\models_{\mathfrak{M}_{n>2}} = \models_{\mathfrak{M}_2}$ (ou, o que é o mesmo, $Cn_{\mathfrak{M}_{n>2}} = Cn_{\mathfrak{M}_2}$).

1.3.3 Completude funcional

Seja agora que $n \geq 2$, $n \in \mathbb{N}$; denotamos por E_n o conjunto $\{1, 2, ..., n\}$ e por $\mathcal{U}_n$ uma álgebra qualquer da forma $(E_n, f_1, ..., f_m)$. Em especial, denotamos o conjunto de todos os mapeamentos m-ários definidos em E_n com valores no mesmo conjunto por

$$Z_n^m = \{f \mid f : E_n^m \longrightarrow E_n\}, \quad m \geq 0, m \text{ finito},$$

e denotamos o conjunto de todos os mapeamentos definidos em E_n com valores em Z_n^m por

$$Z_n = \bigcup_{m \in \omega} Z_n^m$$

para ω o conjunto de todos os números naturais.

Dizemos que uma função $f \in Z_n^m$ depende de um argumento x_i, $1 \leq i \leq m$, sse existem $u, v \in E_n$ tais que $u \neq v$ e

$$f(x_1, ..., x_{i-1}, u, x_{x+1}, ..., x_m) \neq f(x_1, ..., x_{i-1}, v, x_{x+1}, ..., x_m).$$

1.3.12. DEFINIÇÃO. Diz-se de um conjunto de funções $X \subseteq Z_n$ que é *funcionalmente completo* sse pode definir-se qualquer função $f \in Z_n$ por meio das funções em X.

Por outras palavras, pode representar-se o mapeamento finito $f : E_n^m \longrightarrow E_n$ como uma composição das operações $f_1, ..., f_m$. Diz-se então que uma álgebra $\mathcal{U}_n$ é funcionalmente completa (Post, 1921).

Embora seja uma propriedade que devemos estar preparados para dispensar em certas circunstâncias (por exemplo, no caso de lógicas infinitamente multivalentes), a completude funcional de álgebras n-valentes é altamente relevante na medida em que as lógicas proposicionais que se fundam nelas são lógicas de todos os conetores n-valentes extensionais possíveis.

Seguem-se alguns resultados importantes com respeito à completude funcional.

1.3.13. TEOREMA. Existe um algoritmo de decisão para a completude funcional de subconjuntos finitos de Z_n.

Demonstração. (Ver Bolc & Borowik, 2003, p. 24-5.) □

1.3.14. TEOREMA (Post, 1921). Se $\mathcal{U}_n$ é funcionalmente completo para m variáveis, $m \geq 2$, então também é funcionalmente completo para $m+1$ variáveis e é pois também funcionalmente completo.

O teorema 1.3.13 reduz o problema da completude funcional das álgebras multivalentes ao problema da definibilidade de todos os conetores unários e binários: dado n, numa lógica n-valente um qualquer lugar dado na tabela de verdade pode ser ocupado por n valores de verdade. Para um conetor de k lugares, a tabela de verdade tem espaço para

$$\underbrace{n \times n \times ... \times n}_{k \text{ vezes}} = n^k$$

entradas e para cada uma delas haverá uma qualquer de n possibilidades, pelo que podemos ter

$$\underbrace{n \times n \times ... \times n}_{n^k \text{ vezes}} = n^{n^k}$$

possíveis tabelas de verdade de k lugares para uma dada lógica n-valente. Temos pois que o número de conetores unários e binários é de n^n e n^{n^2}, respetivamente.

Post (1921) produziu a primeira álgebra lógica n-valente para $n \geq 2$ mostrando que de modo a estabelecer a completude funcional de $\mathcal{U}_n$ é suficiente a geração de duas funções. Estas são a função de rotação cíclica com um argumento e a função máximo de dois argumentos

$$\neg_n x = \begin{cases} 1 & \text{se } x = n \\ i+1 & \text{se } x = i \neq n \end{cases}$$

e

$$x \vee y = max(x, y)$$

para a negação e a disjunção, respetivamente.

Logo, toda a álgebra de Post n-valente

$$\mathcal{P}_n = (E_n, \neg_n, \vee)$$

é funcionalmente completa.

Existem ainda outros critérios de completude que podem ser úteis. Por exemplo,

1.3.15. Teorema (Słupecki, 1939a). Para $n \geq 2$, n finito, uma álgebra n-valente $\mathcal{U}_n$ é funcionalmente completa sse nela se pode definir

1. todas as operações de um argumento em E_n;
2. pelo menos uma operação de dois argumentos $f(x, y)$ cujo contradomínio consiste em todos os valores i para $1 \leq i \leq n$.

1.3.16. Teorema. A álgebra booleana de dois elementos

$$\mathcal{B}_2 = (\{0, 1\}, \vee, \wedge, \neg, 1, 0),$$

da qual $\mathcal{P}_2 = (E_2, \neg_2, \vee)$ é um reduto, é funcionalmente completa.

1.4 Exercícios

1. Determine o valor de verdade das seguintes fórmulas pela construção das respetivas tabelas de verdade:

(a) $\neg(((A \leftrightarrow B) \leftrightarrow (\neg C \wedge D)) \vee \neg E)$; (b) $(\neg(P \vee Q) \wedge \neg(R \vee S)) \rightarrow (R \rightarrow \neg S)$;
(c) $((P \vee \neg(R \wedge S)) \rightarrow (Q \vee T)) \vee (\neg(P \wedge \neg S))$;(d) $(A \rightarrow (B \wedge C)) \rightarrow (A \rightarrow C)$;
(e) $\phi \leftrightarrow \neg(\phi \vee \psi)$

2. Verifique se as seguintes deduções são válidas em LPC (1) utilizando tabelas de verdade e (2) com o sistema axiomático do exemplo 1.2.17:

(a) $A \leftrightarrow B, C \leftrightarrow D \vdash A \leftrightarrow D$; (b) $\neg A \vee B \vdash C$; (c) $\neg A \vdash A \rightarrow \neg A$;
(d) $A \wedge B \vdash B \leftrightarrow A$; (e) $\neg\phi \rightarrow \phi \vdash \phi$; (f) $\phi \wedge \psi \models \phi \rightarrow \psi$;
(g) $\neg P \vee Q \models \neg(P \wedge \neg Q)$

3. Utilizando o sistema axiomático LPCA e regras derivadas demonstre os seguintes teoremas de LPC:

(a) $\vdash A \rightarrow A$; (b) $\vdash \neg(A \leftrightarrow \neg A)$; (c) $\vdash \psi \rightarrow (\phi \vee \neg\phi)$
(d) $\vdash (P \wedge Q) \vee (\neg P \vee \neg Q)$

4. Formalize as seguintes asserções em LPOC:

(a) Toda a gente gosta de alguma coisa.
(b) O Pedro gosta de toda a gente.
(c) Nem toda a gente gosta do Pedro.
(d) Há quem não goste do Pedro.
(e) Algumas pessoas não gostam de si mesmas.
(f) Há algo do qual toda a gente gosta.
(g) Há algo do qual ninguém gosta.
(h) Há algo do qual nem o Pedro nem o Rui gostam.
(i) Algumas pessoas gostam daqueles que gostam delas.
(j) Algumas pessoas não gostam daqueles que não gostam delas.
(l) Ninguém gosta do Rui.
(m) Se alguém gosta do Rui, então detesta batatas.
(n) Nenhum humano é imortal.

5. Seja o domínio $\mathscr{D}$ o conjunto de todas as pessoas e defina-se $A(x,y)$ como "x ama y". Traduza a seguinte frase em LPOC para português:

$$(\forall x \exists y A(x,y) \wedge \neg \exists x \forall y A(x,y)) \vee (\exists x \forall y (x,y) \wedge \exists x \forall y \neg A(x,y))$$

6. Seja o domínio $\mathscr{D}$ o conjunto de todas as pessoas e de todos os tempos (i.e., momentos) e defina-se $E(x,y,t)$ como "x pode enganar y no momento t". Traduza a seguinte frase em LPOC para português:

$$\forall x (\exists y \forall t E(x,y,t) \wedge \forall y \exists t E(x,y,t) \wedge \exists y \exists t \neg (x,y,t))$$

7. Diga se a seguinte afirmação é verdadeira ou falsa e justifique com exemplos: Uma lógica é de ordem superior se permite a quantificação de conjuntos ou se permite que conjuntos sejam elementos de outros conjuntos. (Nota: Podemos identificar conjuntos com predicados no sentido em que, por exemplo, o predicado $P(x,y)$ é verdadeiro sse $x, y \in P$.)

8. Para cada fórmula, indique qual a ordem dos seus constituintes e finalmente a ordem da fórmula. (Nota: $f(x)$ é um termo, mas f é uma função, ou seja, um conjunto ou um predicado no sentido em que por exemplo a função $f(x) = 2x$ corresponde ao conjunto $f = \{(x, 2x) \mid x \in \mathbb{N}\}$; ver exercício anterior.)

(a) $P(x)$; (b) $P(x) \wedge Q(P)$; (c) $\forall x (P(x))$; (d) $P(x, f, f(x))$;
(e) $\forall x \exists S \exists T \exists q (S(x, q(x)) \wedge T(S))$

9. Represente a seguinte proposição em lógica de segunda ordem: Existe um conjunto S de números naturais $\mathbb{N}$ que não contém 5.

10. Transforme cada uma das seguintes fórmulas num conjunto de cláusulas:

(a) $(A \wedge B) \vee \neg (C \vee D)$; (b) $(P \wedge Q) \rightarrow ((R \wedge S) \rightarrow P)$; (c) $\exists y \forall x (P(x,y) \rightarrow Q(x))$;
(d) $\exists y \forall x P(x,y) \rightarrow Q(x)$; (e) $\forall x \forall y (P(x,y) \vee \exists z Q(x,y,z))$;
(f) $\forall w \exists u \exists v ((\neg P(w,u) \wedge Q(w,v)) \rightarrow R(w,u,v))$

2 Lógicas multivalentes

2.1 Algumas notas históricas

É interessante verificar que Aristóteles, tido comummente como o criador da lógica bivalente, foi também o primeiro a questionar o estatuto bivalente (ou seja, verdadeiro ou falso; consequentemente, não simultaneamente verdadeiro e falso) das proposições acerca de acontecimentos futuros. Com efeito, pode ler-se no seu *De interpretatione* (IX, 19a30):

> Uma batalha naval deverá ou ter lugar ou não ter lugar amanhã, mas não é necessário que deva ter lugar amanhã, tal como também não é necessário que não deva ter lugar, e contudo é necessário que deve ou não ter lugar amanhã.

Estas proposições parecem ser nem necessariamente verdadeiras nem necessariamente falsas, o que desafia tanto o princípio da bivalência como os pilares da lógica clássica, em especial a lei do terceiro excluído, de acordo com a qual para toda a proposição, ou é verdadeira, ou é-o a sua negação. Aristóteles parece com efeito conceber um terceiro valor de verdade, intermediário, que se pode exprimir como "indeterminado".

Conhecido geralmente como o *problema dos futuros contingentes*, este foi um tópico muito debatido na lógica medieval, mas o verdadeiro progresso no estabelecimento de lógicas multivalentes que abordavam diretamente o problema só se deu nos inícios do séc. XX, nomeadamente com J. Łukasiewicz, após os desenvolvimentos fundamentais de formalização na lógica matemática da autoria de Boole e Frege. Estes desenvolvimentos permitiram ainda que preocupações mais (meta)matemáticas fossem formuladas e abordadas numa perspectiva lógica multivalente, como foi o caso de E. Post (Post, 1921) e mais tarde de S. Kleene (Kleene, 1938; 1952). Embora estes dois problemas continuem a estar na base das motivações contemporâneas para lógicas multivalentes, estas cresceram em quantidade e diversidade: parece com efeito não só haver *lacunas de valor de verdade*,[1] como no caso dos futuros contin-

[1] *Truth-value gaps*, em inglês.

gentes, mas ainda *excessos de valor de verdade*,[2] ou contradições verdadeiras (por ex.: leis inconsistentes; paradoxos de auto-referência), aspetos que levantam importantes questões filosóficas com repercussões legais e científicas (Priest, 2008); a estas juntam-se preocupações mais (meta)matemáticas, muitas das quais têm a ver cada vez mais com propriedades computacionais e com (aplicações práticas da) ciência dos computadores em geral (ex.: Bolc & Borowik, 2003; Epstein, 1993; Fitting & Orłowska, 2003; Karpenko, 2006).

Seja como for, as lógicas multivalentes são hoje elementos essenciais da lógica matemática e muitas delas têm formalizações adequadas, embora alguns problemas interpretativos se mantenham em aberto. Começamos a nossa discussão das lógicas multivalentes mais relevantes precisamente com estes últimos.

2.2 Multivalência e problemas de interpretação

2.2.1 A Tese de Suszko e a multivalência

Recorde-se da secção 1.3 que uma interpretação para uma qualquer linguagem $\mathcal{L}^{Prop} = (F, O_1, ..., O_m)$ pode ser dada por uma estrutura de interpretação

$$\mathcal{A} = (\mathscr{A}, f_1, ..., f_m)$$

em que $\mathscr{A}$ é o contradomínio dos correlatos semânticos de $\mathcal{L}^{Prop}$. Isto mostra-se muito conveniente quando $|W| > 2$, uma vez que através do homomorfismo $Hom\left(\mathcal{L}^{Prop}, \mathcal{A}\right)$ pode-se fazer os correlatos semânticos de proposições (i.e., factos e situações) corresponder aos valores lógicos propriamente ditos, os quais se acredita comummente ser apenas dois, a saber, a *verdade* e a *falsidade* (ou seja, o contradomínio de $\mathscr{A}$ pode ser reduzido a $\{0, 1\}$; ver Fig. 2.1); pela mesma razão isto é, porém, altamente problemático, pois parece defender a noção que a lógica "propriamente dita" é bivalente, pondo assim em questão o estatuto dos sistemas lógicos multivalentes. Obviamente, o problema não se põe com W_2, dado que neste caso o correlato semântico em $\mathcal{A}_2$ de uma fórmula em $\mathcal{L}_2^{Prop}$ é necessariamente verdadeiro ou falso. A interpretação semântica nas lógicas multivalentes é um problema em aberto na lógica contemporânea que pode de facto ter impacto na automatização da dedução destas lógicas. Como tal, impõem-se algumas considerações acerca desta questão.

Para começar, fazemos notar, por um lado, que o teorema da completude (teorema 1.3.9) exprime o facto que toda a lógica tarskiana tem uma semântica multivalente;

[2] *Truth-value gluts*, em inglês.

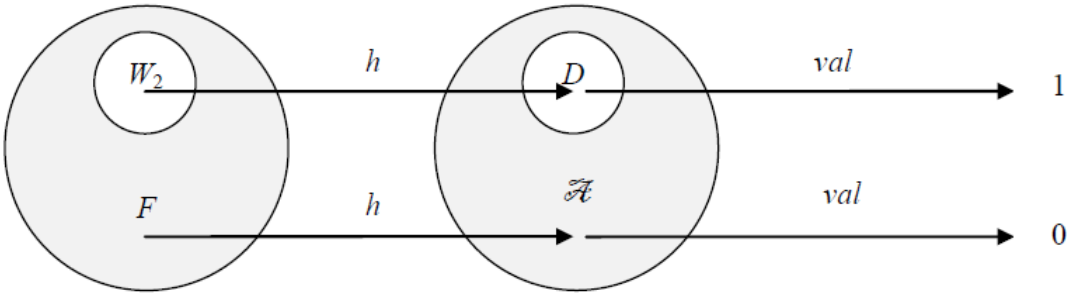


Figura 2.2.1: Relação entre a interpretação homomórfica $h \in Hom\left(\mathcal{L}^{Prop}, \mathcal{A}\right)$ e a valoração lógica $val_h : F \longrightarrow W_2$.

por outro lado, R. Suszko, entre outros, mostrou que toda a semântica multivalente pode ser reduzida a uma semântica bivalente e barafustou contra a "multiplicação de valores lógicos" (Suszko, 1977). Com efeito, dada uma linguagem $\mathcal{L}^{Prop}$ e uma matriz $\mathfrak{M} = (\mathcal{A}, D)$ para a mesma, definimos o conjunto de atribuições $VW_{\mathfrak{M}} = \left\{val_h \mid h \in Hom\left(\mathcal{L}^{Prop}, \mathcal{A}\right)\right\}$ em que

$$val_h(\phi) = \begin{cases} 1 & \text{se} \quad h(\phi) \in D \\ 0 & \text{se} \quad h(\phi) \notin D \end{cases} .$$

Consequentemente,

$$X \models_{\mathfrak{M}} \phi \text{ sse, para toda } val_h \in VW_{\mathfrak{M}}, val_h(\phi) = 1 \text{ sempre que } val_h(X) \subseteq \{1\} .$$

Repetindo-se a definição de atribuições lógicas com respeito a uma qualquer operação de consequência estrutural Cq fazendo uso do *bundle* de Lindenbaum obtém-se o importante resultado que toda a lógica estrutural $\left(\mathcal{L}^{Prop}, Cq\right)$ é logicamente bivalente.

A posição de Suszko de acordo com a qual há apenas dois valores lógicos, a saber, verdadeiro e falso, é conhecida de modo mais ou menos informal como a *Tese de Suszko*, e o seu autor empenhou-se em tornar esta tese não trivial distinguindo os dois valores de verdade *lógicos* dos valores de verdade *algébricos*, os quais têm apenas um papel referencial e podem ser em número infinito. Nesta perspetiva, as lógicas multivalentes são obviamente também bivalentes, mas para $|W| > 2$ elas são somente *referencialmente* multivalentes (vs. *logicamente* multivalentes). Ora, dá-se que de acordo com o *Axioma de Frege* (prop. 1.3.1) duas proposições de LPC com o mesmo valor de verdade descrevem o mesmo objeto ou o mesmo facto, ou seja, têm a mesma referência ou a mesma denotação; consequentemente, do ponto de vista da

lógica clássica o valor de verdade lógico é o único atributo de uma proposição que interessa considerar.[3] Nesta perspetiva, os valores lógicos e os valores referenciais coincidem em LPC, mas são obviamente distintos nas lógicas multivalentes.

É provável que as preocupações de Suszko não incluam as múltiplas aplicações das lógicas multivalentes que são hoje essenciais (ver Introdução), razão pela qual ele parece tão empenhado em eliminá-las do panorama lógico, mas ele realçou o facto que um sistema lógico é, num sentido fundamental, um sistema de preservação da verdade por oposição à igualmente "nobre" falsidade. Numa linha de argumentação próxima desta, Rescher (1969) defendeu que, embora as lógicas multivalentes puramente abstratas possam ter aplicações importantes, podem bem não ser mais do que meros cálculos no sentido estrito do termo.

É evidente que a bivalência lógica de uma dada lógica estrutural é uma consequência direta da divisão dos elementos de $\mathscr{A}$ em dois conjuntos complementares de valores designados e não designados, D e $\mathscr{A} - D$, respetivamente, por sua vez uma consequência da construção de matrizes lógicas (ver Fig. 2.1; cf. Malinowski, 2008).

Com isto em mente, tentou-se já construir matrizes não caracterizáveis por conjuntos complementares (por ex., Malinowski, 2012). De facto, seja como for é ainda útil (se não mesmo necessário) em certas circunstâncias trabalhar num contexto clássico bivalente, nomeadamente com a automatização da dedução em lógicas multivalentes em vista, e houve também já muitas tentativas de encontrar reduções bivalentes para (as) lógicas multivalentes (ex.: Caleiro et al., 2007). Mostramos em seguida como isto pode ser feito de modo "natural" através das noções de quase-tautologia (e logo quase-contradição) e quase-derivabilidade.

2.2.2 Verofuncionalidade, tautologia, contradição e derivabilidade em lógicas multivalentes

O mesmo resultado de bivalência clássica pode obter-se pela consideração do princípio de verofuncionalidade que se encontra na raiz da descrição matricial de LPC através da sua generalização por meio do princípio de interpretação na secção 1.3.1. Visto que a maior parte das lógicas proposicionais multivalentes são de facto extensões conservadoras de matrizes de LPC, pode-se determinar as condições-padrão que

[3] Isto expressa-se por meio do conetor da equivalência material em que a identidade no lado direito é a identidade dos valores lógicos (vs. a identidade das proposições)

$$x \leftrightarrow y \in \{1\} \text{ sse } x = y.$$

.

nos permitem caracterizá-las como bivalentes. Adaptamos aqui Rosser & Turquette (1952) de acordo com a notação e definições acima.

2.2.1. PROPOSIÇÃO. Dada uma qualquer matriz da forma

$$\mathfrak{M}_{n,k} = (\mathcal{U}_n, D_k)$$

em que $\mathcal{U}_n = (E_n, f_1, ..., f_m)$, $E_n = \{1, 2, ..., n\}$ e $D_k = \{1, 2, ..., k\}$ para $1 \leq k < n$,$n \in \mathbb{N}$ e $n \geq 2$, $1 \leq k < n$ expressa um grau decrescente de verdade com 1 a referir-se sempre à verdade e n à falsidade, temos as condições-padrão *gerais*. As condições-padrão *específicas* descrevem os conetores que são vistos como padrão: Sejam $\neg, \rightarrow, \vee, \wedge, \leftrightarrow$ funções de uma dada matriz $\mathfrak{M}_{n,k}$ e seja $\{j_1, j_2, ..., j_n\}$ uma família de funções unárias de $\mathcal{U}_n$. Se para todos $x, y \in E_n$ e $i \in \{1, 2, ..., n\}$

$\neg x \in D_k$	sse	$x \notin D_k$
$x \rightarrow y \notin D_k$	sse	$x \in D_k$ e $y \notin D_k$
$x \vee y \in D_k$	sse	$x \in D_k$ ou $y \in D_k$
$x \wedge y \in D_k$	sse	$x \in D_k$ e $y \in D_k$
$x \leftrightarrow y \in D_k$	sse	ou $x, y \in D_k$ ou $x, y \notin D_k$
$j_i(x) \in D_k$	sse	$x = i$

em que j_i são identificadores de valores lógicos, $j_1, ..., j_k$ são asserções e $j_{k+1}, ..., j_n$ negações, então dizemos que os conetores respetivos de $\mathfrak{M}_{n,k}$ satisfazem as condições-padrão. Seja agora que $O = \{\neg, \rightarrow, \vee, \wedge, \leftrightarrow\}$ e $J_n = \{j_1, j_2, ..., j_n\}$: a matriz $\mathfrak{M}_{n,k}$ com um conjunto de funções definíveis $T \subseteq O \cup J_n$ satisfazendo as condições-padrão correspondentes chama-se *T-padrão* e *padrão* se é T-padrão para $T = O \cup J_n$.

Por seu lado, toda a matriz *O-padrão* $\mathfrak{K}_{n,k} = (E_n, \neg, \rightarrow, \vee, \wedge, \leftrightarrow, D_k)$ é epimórfica com respeito à matriz clássica $\mathfrak{M}_2$ por uma mapeamento $h : E_n \longrightarrow \{0, 1\}$ definido por

$$h(x) = \begin{cases} 1 & \text{se} \quad x \in D_k \\ 0 & \text{se} \quad x \notin D_k \end{cases} .$$

Um **epimorfismo** é um homomorfismo definido por um mapeamento sobrejetivo. Uma função $f : X \longrightarrow Y$ é sobrejetiva sse $f(x) = y$, i.e., se todo $x \in X$ possui um elemento correspondente $y \in Y$.

Segue-se o resultado:

2.2.2. Resultado.

$$E(\mathfrak{K}_{n,k}) = E(\mathfrak{M}_2) = TAUT.$$

Rosser e Turquette mostraram que o conteúdo $E(\mathfrak{M}_{n,k})$ de toda a matriz $\{\rightarrow\} \cup J_n$-padrão $\mathfrak{M}_{n,k}$ é axiomatizável por meio do conjunto seguinte de axiomas e das regras MP e SUB:

(A1) $(B \rightarrow (A \rightarrow B))$
(A2) $(A \rightarrow (B \rightarrow C)) \rightarrow (B \rightarrow (A \rightarrow C))$
(A3) $(A \rightarrow B) \rightarrow ((B \rightarrow C) \rightarrow (A \rightarrow C))$
(A4) $(j_i(A) \rightarrow (j_i(A) \rightarrow B)) \rightarrow (j_i(A) \rightarrow B)$
(A5) $(j_n(A) \rightarrow B) \rightarrow ((j_{n-1}(A) \rightarrow B) \rightarrow (... \rightarrow ((j_1)(A) \rightarrow B) \rightarrow B)...)$
(A6) $j_i(A) \rightarrow A$ em que $i = 1, 2, ..., k$
(A7) $j_{i(r)}(A_r) \rightarrow \left(j_{i(r-1)}(A_{r-1}) \rightarrow \left(... \rightarrow j_{i(1)}(A_1) \rightarrow j_f(O(A_1, ..., A_r))\right)...\right)$ em que $f = f(i(1), ..., i(r))$, f é uma qualquer função de $\mathfrak{M}_{n,k}$ e O um conetor proposicional associado a f

Apesar destes resultados, é evidente que a existência de valores de verdade múltiplos põe seriamente em causa as importantes noções de tautologia e contradição (def.s 1.1.20 e 1.1.22) e, por reflexão, a noção fundamental de derivabilidade, se aquelas são imediatamente generalizáveis (como de facto o são) de modo a serem aplicadas em sistemas lógicos multivalentes. É fácil de ver que se num sistema lógico multivalente S_n qualquer atribuição de valor de verdade é sempre tal que difere de 0 e 1 (ou F e V), então o sistema em causa não tem nem tautologias nem contradições no sentido clássico.

Contudo, pode-se "mitigar" este problema procedendo do modo que a seguir se apresenta.

2.2.3. Definição. Sejam dados um conjunto $W_n = \{v_0, v_1, ..., v_{n-1}\}$ de valores de verdade para $n > 2$ e um conjunto de valores designados $D_k \subset W_n$. (i) Diz-se de uma fórmula invariavelmente tomando (ou podendo tomar) um qualquer $v_j \in D_k$ para toda e qualquer atribuição de valores de verdade para as suas variáveis proposicionais que é uma *quase-tautologia*. (ii) Diz-se de uma fórmula invariavelmente tomando (ou podendo tomar) um qualquer $v_i \notin D_k$ para toda e qualquer atribuição de valores de verdade para as suas variáveis proposicionais que é uma *quase-contradição*.

Dado, porém, que um conjunto de valores *não designados* não é necessariamente estabelecido como $W_n - D_k$, $1 \leq k < n$, (ou seja, um conjunto de valores de verdade não designados não tem de ser composto pelos valores de verdade que restam após a seleção dos valores designados; Rescher, 1969), denotamos um conjunto de valores designados por D^+ e um conjunto de valores *antidesignados* por D^-. Podemos então reformular a definição 2.2.3:

2.2.4. DEFINIÇÃO. Sejam dados um conjunto $W_n = \{v_0, v_1, ..., v_{n-1}\}$ de valores de verdade para $n > 2$, um conjunto de valores designados $D^+ \subset W_n$ e um conjunto de valores antidesignados $D^- \subset W_n$. (i) Diz-se de uma fórmula invariavelmente tomando (ou podendo tomar) um qualquer $v_j \in D^+$ para toda e qualquer atribuição de valores de verdade para as suas variáveis proposicionais que é uma *quase-tautologia*. (ii) Diz-se de uma fórmula invariavelmente tomando (ou podendo tomar) um qualquer $v_i \in D^-$ para toda e qualquer atribuição de valores de verdade para as suas variáveis proposicionais que é uma *quase-contradição*.

Falaremos apenas de conjunto de valores designados e denotá-lo-emos simplesmente por D se não considerarmos um qualquer conjunto de valores antidesignados. Fazemos notar que para um qualquer $v_i \in W_n$, $n > 2$, não é necessário que se verifique a condição $v_i \in D^+$ sse $v_i \notin D^-$. Esta estratégia permite-nos equacionar conjuntos de tautologias e de contradições de lógicas multivalentes com aqueles de LPC (Rescher, 1969).

2.2.5. EXEMPLO. Os seguintes conetores são tais que o sistema que definem (B_3; ver abaixo) pode ter apenas tautologias bivalentes e contradições bivalentes; além disso,

$$TAUT\,(\mathrm{B}_3) = TAUT\,(\mathrm{C}_2)$$

$$CONT\,(\mathrm{B}_3) = CONT\,(\mathrm{C}_2)$$

em que $TAUT(\mathrm{X})$ e $CONT\,(\mathrm{X})$ designam respetivamente os conjuntos de tautologias e contradições de um dado sistema lógico X. (Designaremos LPC por C_2.)

A	$\neg A$
+V	F
±I	I
−F	V

	$\wedge$			$\vee$			$\rightarrow$			$\leftrightarrow$		
$A\backslash B$	V	I	F	V	I	F	V	I	F	V	I	F
V	V	I	F	V	I	V	V	I	F	V	I	F
I	I	I	I	I	I	I	I	I	I	I	I	I
F	F	I	F	V	I	F	V	I	V	F	I	V

2.2.6. Definição (quase-verofuncionalidade; Rescher, 1969). Um sistema S de lógica proposicional definido por tabelas de verdade que não são monovalentes mas nas quais valores múltiplos podem ocorrer em pelo menos alguns lugares caracteriza-se como sendo *quase-verofuncional*.

2.2.7. Exemplo (Rescher, 1969). Os conetores seguintes definem um sistema quase-verofuncional:

A	$\neg A$
+V	F
I	I
−F	V

	$\wedge$			$\vee$			$\rightarrow$			$\leftrightarrow$		
$A\backslash B$	V	I	F	V	I	F	V	I	F	V	I	F
V	V	I	F	V	V	V	V	I	F	V	I	F
I	I	I,F	F	V	I,V	I	V	I,V	I	I	I,V	I
F	F	F	F	V	I	F	V	V	V	F	I	V

Evidentemente, tudo isto desafia a generalização da importante noção de derivabilidade clássica às lógicas multivalentes; em especial a noção de validade pede uma reavaliação. Das definições acima de quase-tautologia e quase-contradição segue-se a noção de quase-derivabilidade, mas a noção seguinte é a nosso ver ainda mais relevante na medida em que de um certo modo junta as noções de derivabilidade e quase-derivabilidade (Bergmann, 2008):

2.2.8. Definição (derivabilidade em grau). Num sistema lógico multivalente uma fórmula ϕ é *derivável em grau* de um conjunto de fórmulas Γ se o valor de verdade de ϕ nunca pode ser inferior ao valor mínimo de todas as fórmulas $\psi \in \Gamma$.

2.2.9. Proposição (validade em grau). Um argumento é *válido em grau* num sistema lógico multivalente se a sua conclusão é derivável em grau do conjunto das suas premissas.

Isto é prontamente generalizável às lógicas infinitamente multivalentes:

2.2.10. DEFINIÇÃO (derivabilidade em grau n). Uma fórmula ϕ é *derivável em grau n* de um conjunto de fórmulas Γ se $1-n$, $n \subseteq [0,1]$, é a máxima distância para baixo entre Γ e ϕ.

A noção de *validade em grau n* segue-se imediatamente. Fazemos notar que a derivabilidade em grau 1 (a validade em grau 1) coincide com a derivabilidade em grau (a validade em grau).

Contudo, a escolha de um conjunto de valores designados permite-nos uma definição mais geral de validade em lógicas multivalentes:

2.2.11. DEFINIÇÃO (validade em lógicas multivalentes). Dado um conjunto designado $D \subset W, D \neq \emptyset$ dizemos que uma inferência de uma fórmula A a partir de um conjunto de fórmulas Γ é válida sse preserva valores designados, i.e.,

$$\Gamma \models_D A \quad \text{sse para qualquer interpretação } \mathcal{I}, \text{ sempre que } val_{\mathcal{I}}(B) \in D$$

$$\text{para toda } B \in \Gamma, val_{\mathcal{I}}(A) \in D.$$

2.3 Propriedades estruturais das lógicas multivalentes

Acima abordámos relações *interpretativas* entre C_2 e lógicas multivalentes; introduzimos agora algumas propriedades importantes de lógicas multivalentes que as colocam numa relação *estrutural* com C_2. Seguimos Rescher (1969).

2.3.1. DEFINIÇÃO (normalidade). Numa tabela de verdade multivalorada dita *normal* uma atribuição de valor de verdade clássica comporta-se exatamente como em LPC, ou seja, toda a fórmula que é verdadeira nessa atribuição na lógica multivalente é igualmente verdadeira nessa atribuição em C_2 e toda a fórmula que é falsa nessa atribuição numa lógica multivalente é igualmente falsa nessa atribuição em C_2.

2.3.2. PROPOSIÇÃO. Diz-se de uma lógica multivalente que é uma *lógica normal* se as tabelas de verdade dos seus conetores básicos são normais.

2.3.3. DEFINIÇÃO (uniformidade). Um conetor numa lógica multivalente é *uniforme* se, sempre que ambas as entradas (de uma linha/coluna) na tabela de verdade restritamente clássica concordam no mesmo valor de verdade clássico (ou seja, V ou F) para uma fórmula composta, então a linha/coluna inteira apresenta o mesmo valor de verdade.

2.3.4. PROPOSIÇÃO. Uma lógica multivalente cujas tabelas de verdade dos seus conetores básicos são uniformes chama-se uma *lógica uniforme*.

Apresentamos em seguida uma outra propriedade importante de algumas lógicas multivalentes que se tornará mais clara após a abordagem da lógica trivalente de Kleene:

2.3.5. DEFINIÇÃO (regularidade K). A tabela de verdade de um conetor multivalorado é *K-regular* se nunca se dá o caso que um valor de verdade clássico é atribuído quando a um dos constituintes é atribuído o valor de verdade I (indeterminado), a menos que o valor de verdade clássico ocorra uniformemente numa linha ou coluna da tabela.

2.3.6. PROPOSIÇÃO. Uma lógica multivalente é uma *lógica K-regular* quando as tabelas de verdade dos seus conetores básicos são K-regulares.

2.3.7. DEFINIÇÃO (decisividade). A tabela de verdade de um conetor proposicional de um sistema lógico S_n, $n \geq 2$, é *decisiva* se, para o seu conjunto de fórmulas F_S se verifica que $val_S : F_S \longrightarrow W_2$.

2.3.8. RESULTADO. Obviamente, C_2 é decisivo.

2.4 Lógicas proposicionais multivalentes

2.4.1 Lógicas de Łukasiewicz

2.4.1.1 A lógica proposicional trivalente $\mathrm{Ł}_3$ de Łukasiewicz e o sistema axiomático $\mathrm{Ł}_3\mathrm{A}$

Começamos por introduzir a lógica proposicional trivalente de Łukasiewicz, denotada por $\mathrm{Ł}_3$. Isto permitir-nos-á importantes generalizações com respeito a lógicas quer finita quer infinitamente multivalentes.

Os trabalhos em lógica multivalente do lógico polaco Jan Łukasiewicz datam de pelo menos 1918, sendo no período entre 1920 e 1930 (Łukasiewicz, 1920; 1930) que o seu sistema multivalente conhecido como $Ł_3$ se estabeleceu. Este sistema tem três valores de verdade que designaremos por F, I e V, abreviações para "falso", "nem verdadeiro nem falso" (ou seja, *intermédio* ou *indeterminado*) e "verdadeiro", respetivamente. (Łukasiewicz concebeu originalmente o conjunto de valores de verdade $\{0, 1/2, 1\}$ e usá-lo-emos sempre que necessário.)

O valor de verdade I (originalmente: $1/2$) foi motivado por proposições contingentes respeitantes ao futuro como, por exemplo, "Estarei em Varsóvia dentro de um ano", as quais parecem ser *nem verdadeiras nem falsas* (uma lacuna de valor de verdade; cf. secção 2.1) no momento em que são proferidas, pois é possível, embora não necessário, que o seu autor esteja de facto em Varsóvia na altura indicada. Com efeito, se a proposição for verdadeira no momento em que é proferida, então é necessária, o que contradiz a suposição; por outro lado, se for falsa, então é impossível que o seu autor esteja em Varsóvia na altura referida, o que de igual modo contradiz a suposição. Assim sendo, Łukasiewicz concluiu que se justificava a existência de um terceiro valor de verdade representando "o possível" e juntando "o verdadeiro" e "o falso" (cf. Łukasiewicz, 1930 (ver trad.: 1967, p. 53)).

Com esta interpretação da contingência futura em mente, Łukasiewicz introduziu operadores modais para a possibilidade e para a necessidade, os quais denotamos por $\Diamond$ e $\Box$, respetivamente, definidos pela seguinte tabela de verdade:

A	$\Diamond A$	$\Box A$
V	V	V
I	V	F
F	F	F

É óbvio que se eliminarmos o valor de verdade I desta tabela a introdução dos operadores modais torna-se trivial, com "$A \leftrightarrow \Diamond A$", "$\Box A \leftrightarrow A$" e "$\Box A \leftrightarrow \Diamond A$" apresentando-se como tautologias.[4]

De modo a construir $Ł_3$, Łukasiewicz considerou como primitivos os conetores $\neg$ e $\rightarrow$ e, estendendo a sua interpretação clássica, propôs as seguintes tabelas de verdade para $\neg_{L_3}$ e $\rightarrow_{L_3}$ (conetores e martelos com subíndices denotando sistemas lógicos só serão usados para desambiguar):

[4]Malinowski (1993), p. 19ss, desenvolve este tópico.

A	$\neg A$
V	F
I	I
F	V

$\rightarrow$	V	I	F
V	V	I	F
I	V	V	I
F	V	V	V

Os três conetores lógicos restantes definem-se do seguinte modo:

$$\begin{array}{lcl} A \vee B & := & (A \rightarrow B) \rightarrow B \\ A \wedge B & := & \neg(\neg A \vee \neg B) \\ A \leftrightarrow B & := & (A \rightarrow B) \wedge (B \rightarrow A) \end{array}$$

As suas tabelas de verdade apresentam-se como se segue:

$\vee$	V	I	F
V	V	V	V
I	V	I	I
F	V	I	F

$\wedge$	V	I	F
V	V	I	F
I	I	I	F
F	F	F	F

$\leftrightarrow$	V	I	F
V	V	I	F
I	I	V	I
F	F	I	V

Wajsberg (1931) mostrou que o fragmento $(\neg, \rightarrow)$ de Ł$_3$ é axiomatizável naquilo que designaremos por Ł$_3$A (i.e., Ł$_3$A é um sistema axiomático correto e completo para Ł$_3$), constituído pelos axiomas Ł$_3$1-Ł$_3$4 abaixo e pelas regras de inferência MP e SUB:

$$\begin{array}{ll} (Ł_3 1) & A \rightarrow (B \rightarrow A) \\ (Ł_3 2) & (A \rightarrow B) \rightarrow ((B \rightarrow C) \rightarrow (A \rightarrow C)) \\ (Ł_3 3) & (\neg B \rightarrow \neg A) \rightarrow (A \rightarrow B) \\ (Ł_3 4) & ((A \rightarrow \neg A) \rightarrow A) \rightarrow A \end{array}$$

É evidente, dadas as definições acima dos restantes conetores de Ł$_3$, que Ł$_3$A se estende à totalidade de Ł$_3$. Bergmann (2008) dá uma descrição completa de Ł$_3$A, do qual enunciamos um resultado importante (dois outros resultados importantes serão enunciados abaixo):

2.4.1. Resultado. Qualquer derivação em LPCA que não envolva LPC2 é uma derivação em Ł$_3$A e todo o axioma que se pode derivar em LPCA sem utilizar LPC2 é um axioma derivado em Ł$_3$A.

Adicionalmente, aumentando o conjunto dos conetores primitivos de Ł$_3$A com T, Słupecki (1936) obteve um sistema axiomático para a lógica trivalente funcionalmente completa. Com efeito, de modo a verificar que a álgebra de três elementos de Łukasiewicz $\mathscr{L}_3 = (\{0, 1/2, 1\}, \neg, \rightarrow, \vee, \wedge, \leftrightarrow)$ não é funcionalmente completa basta ver que a função constante $T : Tx = 1/2$ para todo $x = \{0, 1/2, 1\}$; contudo, e pelo teorema 1.3.15, a adição de T ao conjunto das funções de Ł$_3$ leva a uma álgebra de três elementos funcionalmente completa; consequentemente, adicionando os seguintes dois axiomas a Ł$_3$A obtém-se um sistema axiomático para a lógica trivalente funcionalmente completa:

$$\begin{array}{ll} (Ł_3 5) & TA \rightarrow \neg TA \\ (Ł_3 6) & \neg TA \rightarrow TA \end{array}$$

Uma inspeção das tabelas de verdade dos conetores de Ł$_3$ revela que quando se considera apenas os valores de verdade V e F as tabelas de verdade de cada um dos conetores são exatamente as tabelas de C$_2$. Por outras palavras, sempre que os conetores operam sobre fórmulas com os valores de verdade clássicos, comportam-se exatamente como os conetores correspondentes da lógica clássica.

2.4.2. Resultado. Ł$_3$ é uma lógica normal.

Demonstração. A demonstração é pela inspeção das tabelas de verdade de C$_2$ e de Ł$_3$ nas atribuições dos valores de verdade V e F aos conetores das duas lógicas. □

2.4.3. Resultado. Ł$_3$ é uniforme.

Demonstração. A inspeção das tabelas de verdade de Ł$_3$ dos conetores $\rightarrow$, $\wedge$ e $\vee$ mostra que estes são uniformes. Além disso, o conetor $\leftrightarrow$, que não é uniforme, pode ser definido por meio de $\rightarrow$. Logo, Ł$_3$ é uma lógica uniforme. □

2.4.4. Resultado. Ł$_3$ não é K-regular.

Demonstração. A demonstração é por inspeção das tabelas de verdade de Ł$_3$. □

2.4.1.2 Derivabilidade, tautologias e contradições em Ł$_3$

Dadas as motivações originais de Ł$_3$ (ver acima), justifica-se que a lei do terceiro excluído (T1) não seja válida neste sistema lógico; nem o é o princípio de não contradição (T2). Pelo contrário, a fórmula $A \leftrightarrow \neg A$ não é uma contradição em Ł$_3$. O resultado seguinte é importante:

2.4.5. RESULTADO. Toda a fórmula que é uma tautologia em Ł$_3$ é igualmente uma tautologia em C$_2$ e toda a fórmula que é uma contradição em Ł$_3$ é igualmente uma contradição em C$_2$.

Demonstração. A demonstração segue-se diretamente do facto que Ł$_3$ é normal. □

2.4.6. RESULTADO. Se $\Gamma \models_{Ł_3} A$ então $\Gamma \models A$ (onde $\models$ designa a consequência clássica).

Demonstração. Como acima. □

É porém evidente que o inverso dos resultados 2.4.5-6 não é o caso, ou seja, não se verifica geralmente que (i) toda a fórmula que é uma tautologia (uma contradição) em C$_2$ é igualmente uma tautologia (uma contradição) em Ł$_3$, e (ii) se $\Gamma \models A$ então $\Gamma \models_{Ł_3} A$. Isto mostra que $TAUT\,(Ł_3) \subsetneq TAUT\,(C_2)$ e $CONT\,(Ł_3) \subsetneq CONT\,(C_2)$.

2.4.1.3 Generalizações multivalentes de Ł$_3$

No início dos anos 1930, Łukasiewicz generalizou Ł$_3$ a sistemas tanto finita como infinitamente multivalentes. Um modo conveniente de capturar esta generalização é por meio de uma matriz lógica.

2.4.7. DEFINIÇÃO. Uma matriz lógica da forma

$$\mathfrak{M}_n = (L_n, \neg, \rightarrow, \vee, \wedge, \leftrightarrow, \{1\})$$

para (i) $n \in \mathbb{N}, n \geq 2$ ou (ii) $n = \aleph_0$ ou $n = \aleph_1$ chama-se uma *matriz n-valente de Łukasiewicz* se as condições seguintes são satisfeitas

$$L_n = \begin{cases} \{0, 1/n-1, 2/n-1, ..., 1\} & \text{se } n \in \mathbb{N}, n \geq 2 \\ \{s/w : 0 \leq s \leq w, s, w \in \mathbb{N}, w \neq 0\} & \text{se } n = \aleph_0 \\ [0,1] & \text{se } n = \aleph_1 \end{cases}$$

para a função unária $\neg$ e as funções binárias $\rightarrow, \vee, \wedge, \leftrightarrow$ ($\neg$ e $\rightarrow$ são primitivas) definidas em $Ł_n$ do modo seguinte:

$\neg x = 1 - x$
$x \rightarrow y = min(1, 1 - x + y)$ ou, equivalentemente,

$$x \rightarrow y = \begin{cases} 1 & \text{se } x \leq y \\ 1 - x + y & \text{se } x > y \end{cases}$$

$x \vee y = max(x, y)$
$x \wedge y = min(x, y)$
$x \leftrightarrow y = 1 - |x - y|$

2.4.8. Resultado. Se identificarmos V e F respetivamente com 1 e 0 da série

$$1 = \frac{n-1}{n-1}, \frac{n-2}{n-1}, ..., \frac{2}{n-1}, \frac{1}{n-1}, \frac{0}{n-1} = 0$$

e identificarmos qualquer um dos valores intermédios $\frac{n-2}{n-1}, ..., \frac{2}{n-1}, \frac{1}{n-1}$ com I_i, $0 < i < 1$, facilmente vemos que

(i) $Ł_2 = \mathrm{C}_2$
(ii) $Ł_3 = Ł_3$
(iii) $E(\mathfrak{M}_{n>2}) \subseteq E(\mathfrak{M}_2)$, i.e., $\mathfrak{M}_2$ é uma submatriz de $\mathfrak{M}_{n>2}$.

Uma **subálgebra** de $\mathcal{A} = (X, f_1, ..., f_m)$ é uma álgebra $\mathcal{B} = (Y, g_1, ..., g_m)$ tal que Y é um subconjunto de X fechado com respeito a todas as operações de $\mathcal{A}$ e as operações em $\mathcal{B}$ se definem como restrições a Y das respetivas operações em $\mathcal{A}$. Esta definição estende-se às noções de matriz e **submatriz** lógica.

Podem-se relacionar os conteúdos de matrizes finitas pelo seguinte teorema de Łukasiewicz e Tarski (1930):

2.4.9. Teorema (condição de Lindenbaum). Para finitos $n, m \in \mathbb{N}$, as seguintes condições são equivalentes:

(i) $E(\mathfrak{M}_n) \subseteq E(\mathfrak{M}_m)$
(ii) $m - 1$ é um divisor de $n - 1$, i.e., $(m - 1) \mid (n - 1)$.

Agora com respeito a n infinito, dois importantes resultados são demonstrados por Malinowski (1993) (o resultado 2.4.10 também é demonstrado por Gottwald, 2001):

2.4.10. RESULTADO. $E(\mathfrak{M}_{\aleph_0}) = E(\mathfrak{M}_{\aleph_1})$.

Isto permite-nos designar quer $\mathfrak{M}_{\aleph_0}$ quer $\mathfrak{M}_{\aleph_1}$ simplesmente por $\mathfrak{M}_{\aleph}$.

2.4.11. RESULTADO. $E(\mathfrak{M}_{\aleph_0}) = \cap \{E(\mathfrak{M}_n) : n \geq 2, n \in \mathbb{N}\}$.

Fazendo $n-1 = k(m-1)$, Rescher (1969) generalizou os resultados da condição de Lindenbaum do modo seguinte:

2.4.12. RESULTADO. Para todo o número primo k temos a série

$$E(\mathfrak{M}_{\aleph}) \subset E\left(\mathfrak{M}_{n(k+1)}\right) \subset ... \subset E\left(\mathfrak{M}_{2(k+1)}\right) \subset E\left(\mathfrak{M}_{1(k+1)}\right) \subset (E(\mathfrak{M}_2) = TAUT(\mathrm{C}_2)).$$

Łukasiewicz e Tarski (1930) conjeturaram e subsequentemente Wajsberg (1931) demonstrou que o cálculo proposicional infinitamente multivalente $Ł_{\aleph}$ pode ser axiomatizado, com MP e SUB, do seguinte modo:[5]

$$\begin{array}{ll}
(Ł_{\aleph}1) & A \to (B \to A) \\
(Ł_{\aleph}2) & (A \to B) \to ((B \to C) \to (A \to C)) \\
(Ł_{\aleph}3) & (\neg A \to \neg B) \to (B \to A) \\
(Ł_{\aleph}4) & ((A \to B) \to B) \to ((B \to A) \to A) \\
(Ł_{\aleph}5) & ((A \to B) \to (B \to A)) \to (B \to A)
\end{array}$$

Para $n > 3$, n é finito, $Ł_n$ pode ser axiomatizado pela adição a $Ł_{\aleph}1$ - $Ł_{\aleph}4$ (que para este fim podemos designar por $Ł_n1$-$Ł_n4$) dos seguintes dois axiomas suplementares:

$$\begin{array}{ll}
(Ł_n5) & nA \to (n-1)\,A \\
(Ł_n6) & (n-1)\left((\neg A)^j + (A \cdot (j-1)\,A)\right)
\end{array}$$

[5]Note-se que os três primeiros axiomas são comuns a $Ł_{\aleph}$ e $Ł_3$; $Ł_{\aleph}5$ depende dos outros axiomas.

para $1 < j < n-1$, $j \nmid n-1$ e em que os conetores binários adicionais $+$ e $\cdot$ se definem como $A+B := \neg A \to B$ e $A \cdot B := \neg(A \to \neg B)$, kA é uma substituição da fórmula $\underbrace{A + A + ... + A}_{k \text{ vezes}}$ e A^k é uma substituição da fórmula $\underbrace{A \cdot A \cdot ... \cdot A}_{k \text{ vezes}}$ (Grigolia, 1977).

2.4.2 Os sistemas lógicos multivalentes de Kleene e Bochvar

Embora os sistemas lógicos multivalentes de Kleene e Bochvar tenham na sua origem motivações muito diferentes, podem ser tratados numa mesma secção com o objetivo de apresentar as suas propriedades mais importantes, algumas das quais coincidem. Estes sistemas facilitam ainda uma ponte entre as lógicas de inspiração mais filosófica de Łukasiewicz e as lógicas de Post, as quais têm motivações mais matemáticas.

2.4.2.1 O sistema lógico trivalente de Bochvar

Tal como no caso de Łukasiewicz, a motivação principal de Bochvar foi em grande medida filosófica (mais estritamente: lógica), mas enquanto aquele se preocupou com lacunas de valor de verdade, este mostrou-se mais interessado nos excessos de valor de verdade, ou seja, casos em que uma proposição parece ser simultaneamente verdadeira e falsa. Com efeito, em Bochvar (1939) a conjunção é o conetor primitivo (juntamente com a negação) e é definida como no exemplo 2.2.5 (ignore-se os conjuntos de valores designados e antidesignados). Dadas esta tabela de verdade da conjunção e a definição da negação igual à de $Ł_3$, definem-se os outros conetores do seguinte modo:

$$\begin{array}{lcl} A \vee B & := & \neg(\neg A \wedge \neg B) \\ A \to B & := & \neg(A \wedge \neg B) \\ A \leftrightarrow B & := & (A \to B) \wedge (B \to A) \end{array}$$

Uma inspeção superficial das tabelas de verdade dos conetores da lógica de Bochvar no exemplo 2.2.5 mostra que todos os conetores binários, denotados por O^2 (explicamos o subscrito BI abaixo), apresentam o seguinte padrão:

O^2_{BI}		I	
		I	
I	I	I	I
		I	

Este padrão concretiza a propriedade estrutural definida acima como regularidade K (def. 2.3.5), uma propriedade que tem directamente impacto na existência de tautologias e contradições. De facto, uma fórmula na qual a um dos dois componentes é atribuído o valor de verdade I tem por essa razão o valor de verdade I, tornando assim inoperantes as noções de tautologia e contradição. O terceiro valor de verdade de Bochvar, que designamos por I por razões de uniformidade, é por assim dizer contagioso. Isto explica-se, de um ponto de vista metateórico, pelo facto que a conjunção é um conetor primitivo e que o objeto de Bochvar são os paradoxos semânticos clássicos da lógica matemática, nomeadamente os paradoxos de Russell e de Grelling-Nelson. Em termos mais gerais, uma proposição como "Esta frase é falsa" não pode assumir um único valor de verdade (se verdadeira, esta proposição é falsa; se falsa, é verdadeira), sendo pois *sem sentido*, *paradoxal* ou, na melhor das hipóteses, *indecidível*. De acordo com Bochvar, numa fórmula $\phi = A \wedge B$, sempre que um dos dois conjuntos A ou B assume o valor I, tal deve tornar a conjunção ϕ paradoxal ou sem sentido.

O **paradoxo de Russell**, posto em 1901 por Russell a Frege, tem a seguinte formulação matemática: Seja R o conjunto de todos os conjuntos que não são membros de si próprios, i.e., $R = \{x \mid x \notin x\}$; é fácil de ver que R é um membro de si próprio sse R não é um membro de si próprio (formalmente: $R \in R \leftrightarrow R \notin R$).
A consequência é a inconsistência da teoria dos conjuntos. Em termos mais estritamente lógicos, a noção fregeana de *função* mostra-se inadequada (o Paradoxo de Russell mostrara que uma função não pode ser o seu próprio argumento) e a completa logicização da matemática ambicionada por D. Hilbert mostra-se inexequível.

O **paradoxo de Grelling-Nelson** consta do seguinte: um adjetivo é autológico sse se descreve a si próprio (ex.: *curto*); um adjetivo é heterológico sse não se descreve a si próprio (ex.: *longo*). Obviamente, um adjetivo é ou autológico ou heterológico. O paradoxo põe-se quando se pergunta se o adjetivo *heterológico* é heterológico, uma vez que uma resposta quer afirmativa quer negativa leva a uma contradição.

Por razões que explicaremos abaixo, referir-nos-emos ao reduto acima como o

sistema interno de Bochvar e designá-lo-emos por B_3^I. A inspeção das tabelas de verdade de B_3^I (ver exemplo 2.2.5; ignore-se os sinais $+$, $\pm$ e $-$) produz os seguintes resultados:

2.4.13. RESULTADO. Os conetores de B_3^I são K-regulares e normais, mas não são nem uniformes nem decisivos.

Logo, e nomeadamente graças à regularidade K,

2.4.14. RESULTADO. B_3^I não tem nem tautologies nem contradições.

2.4.15. RESULTADO. Se $\Gamma \models_{BI} A$, então $\Gamma \models A$.[6]

Mas nem toda a derivação de C_2 é uma derivação em B_3^I. Isto mostra que $\models_{BI} \subset \models_2$.

No que respeita o resultado 2.4.14, poder-se-ia aplicar os conceitos de quase-tautologia (quase-contradição) como uma fórmula que nunca assume o valor F (respetivamente, V). Porém, Bochvar tem em mente dois níveis de asserção no mesmo sistema lógico, sendo que além da comum asserção de uma fórmula B como simplesmente B (a *asserção interna*), ele concebe ainda um segundo modo de asserção de acordo com o qual de uma fórmula B se asserta "externamente" que é verdadeira ou falsa. Para tal, concebeu o operador especial de *asserção externa* $a : aB$, caracterizando estes dois modos de asserção por meio das seguintes tabelas de verdade:

B	Asserção externa aB	Asserção interna B
V	V	V
I	F	I
F	F	F

Dado isto, os conetores externos (distinguidos por meio de *) definem-se do modo que se segue, em que $\overset{*}{\neg} A$ se lê "A é falso", $A \overset{*}{\to} B$ lê-se "se A é verdadeiro então B é verdadeiro", $A \overset{*}{\vee} B$ lê-se "A é verdadeiro ou B é verdadeiro", etc.:

[6] Abreviamos B_3^I como BI quando subscrito, uma vez que não há risco de ambiguidade. Faremos o mesmo em casos semelhantes.

$$\begin{array}{rcl} \overset{*}{\neg} A & := & \neg aA \\ A \overset{*}{\vee} B & := & aA \vee aB \\ A \overset{*}{\wedge} B & := & aA \wedge aB \\ A \overset{*}{\rightarrow} B & := & aA \rightarrow aB \\ A \overset{*}{\leftrightarrow} B & := & aA \leftrightarrow aB \end{array}$$

Alguns cálculos básicos produzem as seguintes tabelas de verdade daquilo que chamaremos o *sistema externo de Bochvar* e que designamos por B_3^E:

A	$\overset{*}{\neg} A$
V	F
I	V
F	V

$A \backslash B$	$\overset{*}{\wedge}$ V	$\overset{*}{\wedge}$ I	$\overset{*}{\wedge}$ F	$\overset{*}{\vee}$ V	$\overset{*}{\vee}$ I	$\overset{*}{\vee}$ F	$\overset{*}{\rightarrow}$ V	$\overset{*}{\rightarrow}$ I	$\overset{*}{\rightarrow}$ F	$\overset{*}{\leftrightarrow}$ T	$\overset{*}{\leftrightarrow}$ I	$\overset{*}{\leftrightarrow}$ F
V	V	F	F	V	V	V	V	F	F	V	F	F
I	F	F	F	V	F	F	V	V	V	F	V	V
F	F	F	F	V	F	F	V	V	V	F	V	V

Os resultados que se seguem demonstram-se facilmente:

2.4.16. Resultado. Os conetores de B_3^E são normais e uniformes, mas não K-regulares.

Logo,

2.4.17. Resultado. Os conetores de B_3^E produzem um sistema com tautologias e contradições. Com efeito, temos que

$$TAUT\left(\mathrm{B}_3^{\mathrm{E}}\right) = TAUT\left(\mathrm{C}_2\right)$$

$$CONT\left(\mathrm{B}_3^{\mathrm{E}}\right) = CONT\left(\mathrm{C}_2\right)$$

2.4.18. Resultado. (i) Se $\Gamma \models_{BE} A$, então $\Gamma \models A$; e (ii) se $\Gamma \models A$, então $\Gamma \models_{BE} A$.

Segue-se o importante resultado:

2.4.19. RESULTADO. De certo modo, B_3^E estende C_2 pelo aumento de W_2, i.e., $W_2 \subset W_3^{BE}$, sem que para tal se altere a função clássica $val : F \longrightarrow W_2$, i.e.,

$$val_{BE} : F_{BE} \longrightarrow W_2.$$

Segue-se o resultado óbvio:

2.4.20. RESULTADO. Os conetores de B_3^E são decisivos.

Temos assim que as tautologias e as contradições, bem como a derivabilidade, coincidem nos casos C_2 e B_3^E. Em termos matriciais, a razão para tal é que a matriz

$$\mathfrak{B}_3^E = \left\{ \{\mathrm{V}, \mathrm{I}, \mathrm{F}\}, \overset{*}{\neg}, \overset{*}{\rightarrow}, \overset{*}{\vee}, \overset{*}{\wedge}, \overset{*}{\leftrightarrow}, \{\mathrm{V}\} \right\}$$

identifica (ou não distingue) I e F, enquanto as suas funções se comportam de modo clássico no que respeita a V e F.

Para concluir, enfatizamos que $\mathfrak{B}_3^E$, tal como

$$\mathfrak{B}_3^I = \{\{\mathrm{V}, \mathrm{I}, \mathrm{F}\}, \neg, \rightarrow, \vee, \wedge, \leftrightarrow, \{\mathrm{V}\}\},$$

é um reduto da matriz de Bochvar

$$\mathfrak{B}_3 = \left\{ \{\mathrm{V}, \mathrm{I}, \mathrm{F}\}, \neg, \rightarrow, \vee, \wedge, \leftrightarrow, \overset{*}{\neg}, \overset{*}{\rightarrow}, \overset{*}{\vee}, \overset{*}{\wedge}, \overset{*}{\leftrightarrow}, \{\mathrm{V}\} \right\}.$$

Isto realça o facto que o sistema lógico trivalente de Bochvar é deveras um único sistema, o qual designamos por B_3. Contudo, convém ter em mente que somente

$$E\left(\mathfrak{B}_3^E\right) = E\left(\mathfrak{M}_2\right).$$

2.4.2.2 A lógica trivalente de Kleene

Como se disse acima, a lógica trivalente de Kleene (Kleene, 1938; 1952) tem uma motivação mais claramente matemática do que a de Bochvar; mais especificamente, foi pensada em grande medida com o objetivo de permitir a análise (lógica) de funções parciais. Por exemplo, dada a propriedade P tal que

$$P(x) \quad \text{sse} \quad 1 \leq \frac{1}{x} \leq 2, \quad x \in \mathbb{R}$$

temos que

$$P(x) = \begin{cases} \mathrm{V} & \text{se } \frac{1}{2} \leq x \leq 1 \\ \mathrm{I} & \text{se } x = 0 \\ \mathrm{F} & \text{nos casos restantes} \end{cases},$$

em que I (originalmente: u) claramente tem o significado de *indefinido*, ou *indeterminado* no sentido em que não se pode determinar se $P(x)$ é verdadeiro ou falso, seja porque não há para tal nenhum algoritmo, seja até mesmo porque é irrelevante para a questão em consideração.

Em Kleene (1938), as tabelas de verdade de K_3^S (o S sobrescrito tornar-se-á claro em breve) são definidas do modo seguinte:

A	$\neg A$
V	F
I	I
F	V

	$\wedge$			$\vee$			$\rightarrow$			$\leftrightarrow$		
$A \backslash B$	V	I	F	V	I	F	V	I	F	V	I	F
V	V	I	F	V	V	V	V	I	F	V	I	F
I	I	I	F	V	I	I	V	I	I	I	I	I
F	F	F	F	V	I	F	V	V	V	F	I	V

A inspeção das tabelas de verdade dos conetores de K_3^S produz os resultados seguintes:

2.4.21. Resultado. Os conetores de K_3^S são normais e uniformes, mas não são K-regulares (exceto $\leftrightarrow$).

Dada a matriz

$$\mathfrak{K}_3^S = \{\{\mathrm{V}, \mathrm{I}, \mathrm{F}\}, \neg, \rightarrow, \vee, \wedge, \leftrightarrow, \{\mathrm{V}\}\}$$

é evidente que

$$E\left(\mathfrak{K}_3^S\right) = \emptyset.$$

Isto leva obviamente a resultados indesejados como, por exemplo, as tautologias clássicas $A \rightarrow A$ e $A \leftrightarrow A$ deixarem de o ser em K_3^S. Em Kleene (1952), os conetores acima são designados como sendo *fortes* (*strong*), e novos conetores, desta feita *fracos* (*weak*), para a implicação material, a disjunção e a conjunção são introduzidos (os conetores da negação e da equivalência material mantêm-se inalterados) e definidos do seguinte modo:

$\wedge$ (*)	V	I	F
V	V	I	F
I	I	I	I
F	F	I	F

$\vee$ (*)	V	I	F
V	V	I	V
I	I	I	I
F	V	I	F

$\rightarrow$ (*)	V	I	F
V	V	I	F
I	I	I	I
F	V	I	V

Temos então o sistema K_3^W.Tal como com os conetores fortes, o objetivo foi fixado tendo em vista funções proposicionais aritméticas, com a diferença que agora a recursão era a propriedade principal a considerar, ou seja, Kleene tinha em mente um procedimento de cálculo efetivo que permitisse decidir acerca do estatuto de proposições aritméticas; se um dos constituintes de uma fórmula complexa recebe o valor de verdade I, então o algoritmo simplesmente falha por "contágio" da indeterminação (em qualquer estádio da computação),[7] tal como sucede em B_3^I; explica-se assim o "K" no termo "regularidade K": com efeito, refere-se a Kleene.

Temos os seguintes resultados importantes com respeito a K_3^W:

2.4.22. Resultado. Os conetores de K_3^W são normais, uniformes e K-regulares.

2.4.23. Resultado. Revisitando as tabelas de verdade e a matriz lógica de B_3^I, temos que

$$E\left(\mathfrak{B}_3^I\right) = E\left(\mathfrak{K}_3^W\right) = \emptyset.$$

Com efeito,

$$O_{BI} = O_{KW}$$

para O_S o conjunto de conetores de um sistema lógico qualquer S.

2.4.3 Lógicas de Post

Formalizações das lógicas de Post (Post, 1921) são especialmente importantes dadas as muitas aplicações a que se prestam. Introduzimo-las sumariamente no Capítulo 1 e desenvolvemos agora o seu tratamento no devido detalhe. O aspeto que distingue as lógicas de Post das outras lógicas multivalentes principais é a negação, naquelas regida pela seguinte tabela de verdade de acordo com a função de rotação cíclica acima introduzida:

[7] Quer isto dizer que Kleene não estava interessado em remediar o problema da ausência de tautologias e de contradições na sua lógica trivalente; ele parecia mais interessado em questões de recursão, o que torna o seu sistema potencialmente relevante para a teoria da computabilidade.

A	$\neg_n A$
1	2
2	3
3	4
$\vdots$	$\vdots$
$n-2$	$n-1$
$n-1$	n
n	1

Tal como nos *Principia mathematica* (Whitehead & Russell, 1910), a negação forma juntamente com a disjunção o par de conetores primitivos nas lógicas de Post. Define-se esta última nos sistemas de Post P_n como, para n finito, $A \vee_n B = min\,(A, B)$,[8] e os restantes conetores definem-se então como se segue:[9]

$$\begin{array}{lcl} A \wedge_n B & := & \neg_n(\neg_n A \vee_n \neg_n B) \\ A \to_n B & := & \neg_n A \vee_n B \\ A \leftrightarrow_n B & := & (A \to_n B) \wedge_n (B \to_n A) \end{array}$$

2.4.24. Definição. Para dado $n \geq 2$ e o conjunto de objetos linearmente ordenados $P_n = \{t_1, t_2, ..., t_n\}$, a estrutura algébrica

$$\mathcal{P}_n = (\{t_1, t_2, ..., t_n\}, \neg_n, \vee_n)$$

chama-se uma *álgebra de Post n-valente* e a matriz $\mathfrak{P}_n$ naturalmente associada a ela

$$\mathfrak{P}_n = (\{t_1, t_2, ..., t_n\}, \neg_n, \vee_n, \{t_n\})$$

é a *matriz de Post n-valente.*[10]

2.4.25. Exemplo. Mostramos as tabelas de verdade dos conetores de P_3 (omitimos o n subscrito):

[8] No caso do sistema infinitamente multivalente de Post $\mathrm{P}_{\aleph_0}$ a disjunção é definida como $A \vee B = max(A, B)$.

[9] Post considerou ainda uma família de lógicas com dois parâmetros P_n^μ, em que o primeiro μ dos n valores de verdade é visto como designado.

[10] Note-se, porém, que para alguns autores se define o conjunto de valores designados para P_n como $D_P = \{1\}$, com n a corresponder à falsidade (ex.: Gottwald, 2001; Rescher, 1969); ver exemplo 2.4.25 e os resultados que se lhe seguem. Outros autores (ex.: Bolc & Borowik, 1992) concebem o valor designado como $D_P = \{n-1\}$, com 0 a corresponder à falsidade.

A	$\neg A$
1	2
2	3
3	1

	$\wedge$			$\vee$			$\rightarrow$			$\leftrightarrow$		
$A\backslash B$	1	2	3	1	2	3	1	2	3	1	2	3
1	3	3	2	1	1	1	1	2	2	3	3	3
2	3	1	2	1	2	2	1	2	3	3	1	2
3	2	2	2	1	2	3	1	1	1	3	2	3

É fácil de verificar que enquanto T1 $\notin E(\mathfrak{P}_3)$, a sua contraparte multivalorada $A \vee \neg A \vee \neg\neg A$ é uma tautologia de P_3. Com efeito,

2.4.26. Resultado.

$$A \vee \neg A \vee \neg\neg A \vee ... \vee \underbrace{\neg\neg...\neg}_{(n-1)\text{ vezes}} A \in TAUT(\mathrm{P}_n).$$

Apresentamos agora as propriedades estruturais de P_n:

2.4.27. Resultado. Fazendo 1 corresponder a V e n a F, $1 < i < n$ para i correspondendo a I, facilmente se verifica com respeito a $\mathrm{P}_{n>2}$ que[11]

(i) só o conetor da disjunção é normal;
(ii) só os conetores da disjunção e da implicação material são uniformes;
(iii) nenhum destes sistemas é K-regular;
(iv) nenhum destes sistemas é decisivo.

É pois óbvio que os conetores da conjunção e da implicação material, tal como o da negação, se comportam de modo muito diferente com relação às suas contrapartes clássicas. Uma consequência disto é que, por exemplo, nem a negação de T2, nem nenhuma fórmula

$$\neg\left(A \wedge \neg A \wedge \neg\neg A \wedge ... \wedge \underbrace{\neg\neg...\neg}_{(n-1)\text{ vezes}} A\right)$$

pertence a $TAUT(\mathrm{P}_n)$.

[11]Note-se, contudo, que Post equipou as suas lógicas não standard P_n com uma interpretação semântica que não interpreta proposições mas sim conjuntos de proposições, o que cria algumas dúvidas quanto ao seu estatuto como lógicas proposicionais (ver Rescher, 1969, p. 54-5; Malinowski, 1993, p. 46).

Embora métodos gerais de axiomatização para as lógicas de Post tenham sido já concebidos (ex.: Słupecki, 1939b), os axiomas propriamente ditos ainda não foram produzidos.

2.4.4 Lógicas difusas

Formalismos difusos como, por exemplo, conjuntos difusos *grosso modo* constituindo ou contribuindo para a *lógica difusa* (LD; *fuzzy logic*, em inglês), foram concebidos para exprimir a *vagueza* na linguagem natural (Zadeh, 1965; 1975). A LD é relevante no contexto das lógicas multivalentes na medida em que, independentemente dos problemas linguísticos e filosóficos que originalmente motivaram a sua criação (ex.: o paradoxo sorites; predicados vagos como por exemplo "jovem", "gordo", "vermelho", etc.), pode ser concebida como uma extensão de LC e das lógicas finitamente multivalentes ao permitir quaisquer valores de verdade no intervalo $[0, 1]$, com 0 correspondendo a (absolutamente) F e 1 a (absolutamente) V.

2.4.28. Proposiçao. Seja que para qualquer valor de verdade $x \in W_{LD}$ temos que $0 \leq x \leq 1$, i.e., $x \in [0, 1]$; então $|W_{LD}| = \aleph_1$ pela cardinalidade do contínuo e pela hipótese do contínuo.

A **cardinalidade do contínuo**, denotada por $|\mathbb{R}|$ ou $\mathfrak{c}$, é o tamanho do conjunto $\mathbb{R}$ dos números reais. De acordo com a **hipótese do contínuo**, formulada em 1878 por G. Cantor, não existe nenhum conjunto S cuja cardinalidade se encontre estritamente entre as cardinalidades dos números inteiros e dos números reais, i.e.,

$$\neg\exists S : \aleph_0 < |S| < 2^{\aleph_0}$$

em que $\aleph_0$ é a cardinalidade do conjunto dos números inteiros e $2^{\aleph_0} = \aleph_1 = \mathfrak{c}$. Note-se que $|\,[0, 1]\,| = \mathfrak{c}$.

2.4.29. Definição (validade em LD). Seja o conjunto de valores designados definido como $D_\varepsilon = \{x \mid x \geq \varepsilon, \varepsilon \in (0, 1]\}$; então dizemos que uma inferência $\Gamma \models_\varepsilon \phi$ é *válida* sse para todo $\psi \in \Gamma$ e para todas as interpretações $\mathcal{I}$, se $val_{\mathcal{I}}(\psi) \geq \varepsilon$, então $val_{\mathcal{I}}(\phi) \geq \varepsilon$.

Qualquer lógica definida deste modo é evidentemente uma lógica multivalente. De facto, rege-se pela definição de consequência lógica que se segue da definição imediatamente acima:

2.4.30. DEFINIÇÃO (consequência lógica em LD). $\Gamma \models_{LD} \phi$ sse para todo $\varepsilon \in (0,1]$, $\Gamma \models_{\varepsilon} \phi$.

Porque a LD tem por objetivo exprimir a natureza contínua de algumas propriedades da linguagem natural, o conjunto de valores de verdade de LD é o intervalo $[0,1]$, sendo logo óbvio que podemos conceber LD (ou um qualquer subconjunto de LD) como idêntico a, ou uma variação de, $Ł_{\aleph_1}$ (ex.: Bellman & Zadeh, 1977). Contudo, seria errado equacionar LD com $Ł_{\aleph_1}$ (ou simplesmente $Ł_{\aleph}$, dado o resultado 2.4.10). Com efeito, $Ł_{\aleph}$ é apenas uma das várias lógicas conhecidas como *lógicas difusas de norma-t*, a classe mais importante de LDs. Convém pois começar por abordar a noção de *normas-t* e a *lógica básica* (LB) cuja construção ela imediatamente permite.

2.4.31. DEFINIÇÃO. Uma *norma-t* (por extenso: *norma triangular*) é uma função $\star : [0,1]^2 \longrightarrow [0,1]$ tal que para todo $x, y, z \in [0,1]$

(TN1)	$x \star y = y \star x$	(comutatividade)
(TN2)	$(x \star y) \star z = x \star (y \star z)$	(associatividade)
(TN3)	$x \leq y$ implica que $x \star z \leq y \star z$	(monotonicidade)
(TN4)	$1 \star x = x$	(identidade)
(TN5)	$0 \star x = 0$	(elemento absorvente)

2.4.32. EXEMPLO (normas-t contínuas fundamentais). Para $x, y \in [0,1]$:

1. Norma-t de Łukasiewicz: $x \star_L y := max(0, x + y - 1)$
2. Norma-t de Gödel: $x \star_G y := min(x, y)$
3. Norma-t produto: $x \star_\Pi y := x \cdot y$

Em termos semânticos, uma norma-t é uma interpretação de um tipo específico de conjunção, &, conhecida como *conjunção forte*, tal que $x \star y = x \& y$. Uma lógica difusa de norma-t determina-se pela seleção de uma função de norma-t contínua para o conetor &, definindo-se os restantes conetores em termos de & do modo que se segue para todo $x, y, z \in [0,1]$:

$x \& y \leq z$ sse $y \leq x \to z$ (o resíduo da norma-t)

$x \to y = 1$ sse $x \leq y$
$\neg x = x \to 0$
$x \wedge y = x\&(x \to y)$ (conjunção fraca)
$x \vee y = ((x \to y) \to y) \wedge ((y \to x) \to x)$

Pode ainda demonstrar-se que $x \wedge y = min(x, y)$ e $x \vee y = max(x, y)$. É importante frisar que se selecionarmos 1 como o único valor designado, de modo que $D_\varepsilon = \{1\}$, então toda e qualquer norma-t contínua define uma lógica contínuo-valente $\mathrm{L}(\star)$. Em especial, a lógica LB (*lógica básica*) caracteriza-se pela importante propriedade que qualquer um dos seus teoremas é logicamente verdadeiro em toda a $\mathrm{L}(\star)$.

2.4.33. Definição (LB). O seguinte sistema axiomático, juntamente com a regra MP, define o sistema LB para as fórmulas A, B, C e para a constante 0 denotando zero:

(LB1) $(A \to B) \to ((B \to C) \to (A \to C))$
(LB2) $(A\&B) \to A$
(LB3) $(A\&B) \to (B\&A)$
(LB4) $(A\&(A \to B)) \to (B\&(B \to A))$
(LB5) $(A \to (B \to C)) \to ((A\&B) \to C)$
(LB6) $((A\&B) \to C) \to (A \to (B \to C))$
(LB7) $((A \to B) \to C) \to$
$(((B \to A) \to C) \to C)$
(LB8) $0 \to A$

Diz-se das normas-t contínuas do exemplo 2.4.32 que são fundamentais porque toda a norma-t contínua é uma combinação delas (Hájek, 1998). Os conetores da implicação material e da negação (como resíduo ou pré-complemento, respetivamente) delas derivados comportam-se do modo que se mostra abaixo, dando origem às várias LDs.

2.4.34. Definição ($\text{Ł}_\aleph$). Para a norma-t de Łukasiewicz pode demonstrar-se facilmente que para todo $x, y \in [0, 1]$,

$$x \to y = \begin{cases} 1 & \text{se} \quad x \leq y \\ 1 - x + y & \text{se} \quad x > y \end{cases}$$

e

$$\neg x = 1 - x$$

Temos então a *lógica de Łukasiewicz* $Ł_\aleph$.

2.4.35. PROPOSIÇÃO. Adicionando-se aos axiomas de LB o axioma

$$\neg\neg A \to A$$

obtém-se um sistema axiomático adequado para a norma-t de Łukasiewicz.

2.4.36. DEFINIÇÃO (LG). Para a norma-t de Gödel, temos que para todo $x, y \in [0, 1]$,

$$x \to y = \begin{cases} 1 & \text{se} \quad x \leq y \\ y & \text{se} \quad x > y \end{cases}$$

e

$$\neg x = \begin{cases} 1 & \text{se} \quad x = 0 \\ 0 & \text{se} \quad x > 0 \end{cases}$$

Esta lógica é conhecida como *lógica de Gödel* (LG).[12]

2.4.37. PROPOSIÇÃO. Adicionando-se a LB o axioma

$$A \to (A\&A)$$

obtém-se um sistema axiomático adequado com respeito à norma-t de Gödel.

2.4.38. DEFINIÇÃO (LΠ). Para a norma-t produto, pode demonstrar-se facilmente que para todo $x, y \in [0, 1]$,

$$x \to y = \begin{cases} 1 & \text{se} \quad x \leq y \\ y/x & \text{se} \quad x > y \end{cases}$$

e

[12]Note-se que em LG & comporta-se exatamente como $\wedge$.

$$\neg x = \begin{cases} 1 & \text{se} \quad x = 0 \\ 0 & \text{se} \quad x > 0 \end{cases}$$

Esta é a lógica conhecida como *lógica produto* (LΠ).

2.4.39 Proposição. Se adicionarmos a LB os seguintes axiomas,

$$\neg\neg C \to (((A\&C) \to (B\&C)) \to (A \to B))$$

e

$$(A \wedge \neg A) \to 0$$

obtemos um sistema axiomático adequado com respeito à norma-t produto.

2.4.40. Proposição. As álgebras LB e os seus casos especiais, a saber, as álgebras MV, as álgebras G e as álgebras Π, são semânticas algébricas para LB, $Ł_{\aleph}$, GL, e LΠ, respetivamente.

2.4.41 Definição (Hájek, 1998). 1. Uma *álgebra LB* é uma estrutura

$$\mathcal{LB} = (\mathscr{A}, \wedge, \vee, \star, \to, 0, 1)$$

em que $\wedge, \vee, \star, \to$ são operações binárias e 0 e 1 são constantes tais que

($\mathcal{LB}1$) $(\mathscr{A}, \wedge, \vee, 0, 1)$ é um reticulado limitado
($\mathcal{LB}2$) $(\mathscr{A}, \star, 1)$ é um monoide comutativo
($\mathcal{LB}3$) $\star$ e $\to$ formam um par adjunto: $c \leq a \to b$ sse $a \star c \leq b$ para todo $a, b, c \in \mathscr{A}$
($\mathcal{LB}4$) $a \wedge b = a \star (a \to b)$
($\mathcal{LB}5$) $(a \to b) \vee (b \to a) = 1$

Um **reticulado** é um par (P, R) em que P é um conjunto parcialmente ordenado (ver abaixo) por R e para cada dois elementos $a, b \in P$ existe um supremo (menor limite superior) e um ínfimo (maior limite inferior) de $\{a, b\}$. Alternativamente, define-se um reticulado como um conjunto parcialmente ordenado para o qual se verifica, para quaisquer elementos a e b, que $a \wedge b = inf\,(a, b)$ e $a \vee b = sup\,(a, b)$. Um reticulado é **limitado** se tem um limite inferior 0 e um limite superior 1 (ou I) tais que para qualquer elemento a se tem que

$$a \vee 1 = 1, a \wedge 1 = a, a \vee 0 = a, a \wedge 0 = 0.$$

Segue-se que todo o reticulado finito é limitado.

Uma **ordem parcial** é uma relação binária R num conjunto P tal que para todo $a, b, c \in P$ se tem que

- aRa (reflexividade);
- se aRb e bRa, então $a = b$ (anti-simetria);
- se aRb e bRc, então aRc (transitividade).

Um conjunto com uma ordem parcial chama-se um **conjunto parcialmente ordenado**.
É fácil de ver que a relação $\leq$ no conjunto de números reais, a relação $\subseteq$ no conjunto potência $\mathscr{P}(A)$ de um dado conjunto A e a relação de divisibilidade $a|b$ (a divide b) no conjunto dos números inteiros positivos são exemplos de ordens parciais.
Seja agora Q um subconjunto de P, P é um conjunto parcialmente ordenado por $\leq$. Temos então as seguintes definições fundamentais:

- Um elemento $M \in P$ é um **majorante** ou **limite superior** de S se para todo $x \in S$ se tem que $x \leq M$. Diz-se então que S é **limitado superiormente**.
- Um elemento $m \in P$ é um **minorante** ou **limite inferior** de S se para todo $x \in S$ se tem que $m \leq x$. Diz-se então que S é **limitado inferiormente**.
- Um elemento $s \in P$ é o **supremo** de S se for o menor dos majorantes, i.e., se para todo $x \in S$ se tem que $x \leq s$ e se $x \leq s'$, então $s \leq s'$.
- Um elemento $i \in P$ é o **ínfimo** de S se for o mairo dos minorantes, i.e., se para todo $x \in S$ se tem que $i \leq x$ e se $i' \leq x$, então $i' \leq i$.
- Seja $M \in P$ um majorante; se $M \in S$ então M é o **máximo** de S.
- Seja $m \in P$ um minorante; se $m \in S$, então m é o **mínimo** de S.

Um **monoide** é uma estrutura algébrica com uma única operação binária associativa (i.e., um semigrupo) e com um elemento de identidade. Se a operação é comutativa, então fala-se de um **monoide comutativo**.

2. Uma álgebra LB chama-se uma *álgebra MV* (i.e., álgebra multivalente) se, para todo $x \in \mathscr{A}$ e onde $x' = x \to 0$,

$$x'' = x.$$

3. Uma álgebra LB é uma *álgebra G* (i.e., uma álgebra de Gödel) se para todo $x \in \mathscr{A}$,

$$x^2 = x \star x = x.$$

4. Uma álgebra LB é uma *álgebra Π* (i.e., uma álgebra produto) se para todo $x, y, z \in \mathscr{A}$,

$$x'' = (((x \star y) \to (z \star y)) \to (x \to y))$$

e

$$x \wedge x' = 0.$$

Terminamos esta secção com algumas propriedades importantes facilmente demonstráveis destes sistemas lógicos:

2.4.42. Resultado.

$$TAUT(\text{Ł}_\aleph), TAUT(\text{LG}), TAUT(\text{L}\Pi) \subsetneq TAUT(\text{C}_2);$$

$$CONT(\text{Ł}_\aleph), CONT(\text{LG}), CONT(\text{L}\Pi) \subsetneq CONT(\text{C}_2).$$

2.4.43. Resultado. Os conetores proposicionais destas lógicas são normais para os valores de verdade 0 e 1.

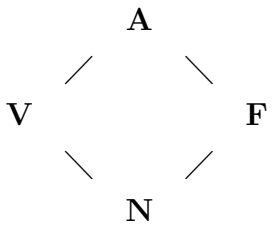
2.4.44. Resultado. $\Gamma \models_{\text{Ł}_\aleph} A, \Gamma \models_{LG} A, \Gamma \models_{L\Pi} A$ sse $\Gamma \models A$, mas o converso não se verifica.

2.4.45. Resultado. Toda a derivação em grau e toda a derivação em grau n destas lógicas (sempre que for o caso) é uma derivação clássica, mas o converso não se verifica.

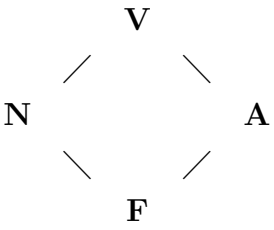
2.4.5 Suplemento: A lógica B_4 de Belnap

Apresentamos sumariamente como suplemento à secção 2.4. uma lógica que, ao contrário das lógicas desta secção, não é uma generalização da lógica clássica.

N. Belnap (1977a, b) criou uma lógica quadrivalente que permite o tratamento de inconsistência na informação com origem em fontes diversas. Para tal, Belnap concebeu o conjunto de valores de verdade $\mathbf{4} = \{\mathbf{V}, \mathbf{F}, \mathbf{A}, \mathbf{N}\}$ que constitui o reticulado de aproximação $\mathbf{A4} = (\mathbf{4}, \sqsubseteq)$,



assim denominado porque a sua ordem "aproxima a informação em", e as tabelas de verdade constituem o reticulado lógico $\mathbf{L4} = (\mathbf{4}, \leq)$,



assim denominado porque a sua ordem é de facto uma ordem lógica, com a conjunção e a disjunção como as operações binárias de *encontro* e *junção*, respetivamente.

Seja o reticulado $(P, \wedge, \vee)$; então $\wedge$ e $\vee$ denotam as operações binárias de **encontro** (*meet*, em inglês) e **junção** (*join*) respetivamente e satisfazem os seguintes axiomas de (1) comutatividade, (2) associatividade e (3) absorção para todo $a, b, c \in P$:

$$(1a) \qquad a \wedge b = b \wedge a$$

$$(1b) \qquad a \vee b = b \vee a$$

$$(2a) \qquad (a \wedge b) \wedge c = a \wedge (b \wedge c)$$

$$(2b) \qquad (a \vee b) \vee c = a \vee (b \vee c)$$

$$(3a) \qquad a \wedge (a \vee b) = a$$

$$(3b) \qquad a \vee (a \wedge b) = a$$

De cada proposição atómica P pode-se dizer que

1. é somente verdadeira (**V**), i.e., $\mathbf{V} = \{\mathrm{V}\}$
2. é somente falsa (**F**), i.e., $\mathbf{F} = \{\mathrm{F}\}$
3. é simultaneamente verdadeira e falsa (**A** [de *ambos*]), i.e., $\mathbf{A} = \{\mathrm{V}, \mathrm{F}\}$
4. tem um estatuto (epistémico) desconhecido (**N**, de *nenhum*), i.e., $\mathbf{N} = \{\}$

pelo que os valores de verdade são mais precisamente "valores de verdade ditos" (*told truth values*). Seguem-se as tabelas de verdade para a negação, a conjunção e a disjunção em B_4:

A	$\neg A$
A	**A**
V	**F**
F	**V**
N	**N**

$\wedge$	**A**	**V**	**F**	**N**
A	**A**	**A**	**F**	**F**
V	**A**	**V**	**F**	**N**
F	**F**	**F**	**F**	**F**
N	**F**	**N**	**F**	**N**

$\vee$	**A**	**V**	**F**	**N**
A	**A**	**V**	**A**	**V**
V	**V**	**V**	**V**	**V**
F	**A**	**V**	**F**	**N**
N	**V**	**V**	**N**	**N**

Fazemos notar que **4** corresponde, enquanto conjunto de valores de verdade propriamente dito, ao conjunto potência de W_2, i.e., $\mathbf{4} = \{\emptyset, \{\mathrm{V}\}, \{\mathrm{F}\}, \{\mathrm{V}, \mathrm{F}\}\}$.

Considere-se a matriz lógica $\mathfrak{B}_4 = (W_4, \neg, \wedge, \vee, D)$ (note-se a ausência do conetor para a implicação material); de acordo com as aplicações em vista, temos $D = \{\{\mathrm{V}\}\}$ ou $D = \{\{\mathrm{V}\}, \{\mathrm{V}, \mathrm{F}\}\}$. Este último conjunto de valores designados faz da lógica de Belnap uma *lógica de relevância* (ou *lógica relevante*), um sistema lógico que impõe que o antecedente e o consequente de uma implicação material apresentem uma relação relevante entre si.[13] Um aspeto central de uma lógica de relevância é a *paraconsistência*, ou seja, o facto que a existência de uma contradição não causa uma "explosão", querendo-se com isto dizer que uma tal lógica rejeita o *princípio (clássico) de explosão* de acordo com o qual do falso tudo se segue (em latim: *ex falso/contradictione sequitur quod libet*), formalmente expresso como $(A \wedge \neg A) \rightarrow B$.

2.5 Quantificação em lógicas multivalentes

Dada a panóplia acima de lógicas multivalentes, é compreensivelmente problemático conseguir-se um método geral de quantificação que sirva para todas elas, nomeadamente um método que generalize a quantificação em LC às lógicas multivalentes. Um tal objetivo torna-se particularmente desejável se se tem em vista métodos gerais de raciocínio automático para estas lógicas. Seja como for, a introdução da quantificação nestas lógicas não é livre de problemas quer numa abordagem mais geral quer numa abordagem mais específica de caso a caso. Apresentamos dois métodos para a generalização de quantificadores para lógicas tanto finita como infinitamente multivalentes e abordamos ainda a quantificação em lógicas difusas.

2.5.1 Quantificadores generalizados

Recorde-se que uma matriz lógica $\mathfrak{M}$ para uma linguagem proposicional $\mathcal{L}^{Prop} = (Vp, O)$ é um triplo $\mathfrak{M} = (W, Fop, D)$ em que W é o conjunto de valores de verdade e $|W| \geq 2$, $Fop = \{f_{O_1}, ..., f_{O_n}\}$ é o conjunto de funções de interpretação e $D \subset W, D \neq \emptyset$ é o conjunto de valores designados. Podemos estender esta a uma matriz lógica de primeira ordem adicionando-lhe um domínio não vazio de quantificação $\mathscr{D}$ e um conjunto $Fq = \{f_{Q_i} \mid Q_i \in Q\}$ para Q o conjunto de quantificadores e onde f_{Q_i} é um mapeamento de subconjuntos de W para W. (Abreviamos f_{Q_i} por $\widetilde{Q_i}$ (def.

[13] Para evitar proposições como "Se estou na lua, então dois e dois são quatro", a lógica de relevância impõe que é necessário (mas não suficiente) que as premissas e a conclusão partilhem fórmulas atómicas (no cálculo proposicional) ou variáveis e constantes (no cálculo de predicados).

1.1.3).)

2.5.1. DEFINIÇÃO. A estrutura $\mathfrak{M}^* = (\mathscr{D}, W, Fop, Fq, D)$ é uma matriz lógica para uma linguagem $\mathcal{L}^{Pred}$ (cf. def. 1.1.14).

Recorde-se ainda a definição 1.1.6, de acordo com a qual o valor de $Q_i x$ é determinado pelo conjunto de instâncias de substituição de uma fórmula A dada uma atribuição val e uma interpretação $\mathcal{I}$, i.e, $val_{\mathcal{I}}(Q_i x A) = Q_i(distr_{\mathcal{I},x} A)$ em que $distr_{\mathcal{I}_x^d} A = \left\{val_{\mathcal{I}_x^d} A \mid d \in \mathscr{D}\right\}$. Podemos agora definir isto por meio da função $\widetilde{Q_i}$ como $val_{\mathcal{I}}(Q_i x A) = \widetilde{Q_i}(\{val_{\mathcal{I}}(A_x(k_d)) \mid d \in \mathscr{D}\})$ para k_d uma constante tal que $val_{\mathcal{I}}(k_d) = d$. A validade mantém-se como na definição 2.2.11. Interessam-nos apenas dois quantificadores, $\forall$ e $\exists$, para os quais, nos termos de LC,

2.5.2. PROPOSIÇÃO. Dado um domínio finito $\mathscr{D} = \{d_1, ..., d_n\}$ de constantes nominais atribuídas a objetos verifica-se que

$$\forall x A(x) \equiv_{\mathscr{D}} A_x(k_{d_1}) \wedge ... \wedge A_x(k_{d_n}) = A(d_1) \wedge ... \wedge A(d_n)$$

e

$$\exists x A(x) \equiv_{\mathscr{D}} A_x(k_{d_1}) \vee ... \vee A_x(k_{d_n}) = A(d_1) \vee ... \vee A(d_n)$$

em que $\equiv_{\mathscr{D}}$ designa a equivalência entre fórmulas numa qualquer interpretação em $\mathscr{D}$.

Isto é naturalmente generalizável às lógicas multivalentes, nas quais tanto a conjunção como a disjunção são operações comutativas e associativas. Visto que nas lógicas multivalentes os valores de verdade se apresentam em graus ou numa sequência ordenada, um modo fácil de generalizar a proposição 2.5.2 é por meio de funções algébricas.

2.5.3. EXEMPLO. Para os sistemas lógicos qŁ$_n$ e qP$_n$, n finito, designando Ł$_n$ e P$_n$ com quantificação, respetivamente, para uma interpretação $\mathcal{I}$ num domínio $\mathscr{D}$,

$$val_{\mathcal{I}}(\forall x A(x)) = min\{val_{\mathcal{I}}(F(d)) \mid d \in \mathscr{D}\}$$

e

$$val_{\mathcal{I}}(\exists x A(x)) = max\{val_{\mathcal{I}}(F(d)) \mid d \in \mathscr{D}\}$$

2.5.4. Exemplo. Para o sistema lógico $\mathrm{qŁ}_{\aleph_1}$, introduzem-se os quantificadores $\forall$ e $\exists$ do seguinte modo:

$$val_{\mathcal{I}}\left(\forall x A(x)\right) = inf\left\{val_{\mathcal{I}}\left(F\left(d\right)\right) \mid d \in \mathscr{D}\right\}$$

e

$$val_{\mathcal{I}}\left(\exists x A(x)\right) = sup\left\{val_{\mathcal{I}}\left(F\left(d\right)\right) \mid d \in \mathscr{D}\right\}$$

Um outro modo de generalizar os quantificadores clássicos às lógicas multivalentes é por meio de descrições.

2.5.5. Exemplo. Para o sistema lógico de Bochvar com quantificação, define-se $\forall$ e $\exists$ como

$$val_{\mathcal{I}}\left(\forall x A(x)\right) = \begin{cases} \mathrm{V} & \text{quando} \quad val_{\mathcal{I}}\left(A\left(d\right)\right) = \mathrm{V} \quad \text{para todo } d \in \mathscr{D} \\ \mathrm{I} & \text{quando} \quad val_{\mathcal{I}}\left(A\left(d\right)\right) = \mathrm{I} \quad \text{para algum } d \in \mathscr{D} \\ \mathrm{F} & \text{caso contrário} \end{cases},$$

e

$$val_{\mathcal{I}}\left(\exists x A(x)\right) = \begin{cases} \mathrm{F} & \text{quando} \quad val_{\mathcal{I}}\left(A\left(d\right)\right) = \mathrm{F} \quad \text{para todo } d \in \mathscr{D} \\ \mathrm{I} & \text{quando} \quad val_{\mathcal{I}}\left(A\left(d\right)\right) = \mathrm{I} \quad \text{para algum } d \in \mathscr{D} \\ \mathrm{V} & \text{caso contrário} \end{cases}.$$

Todos os exemplos acima seguem a generalização proposta por Mostowski (1957). Embora nos exemplos 2.5.3 e 2.5.5 (i.e., para lógicas finitamente multivalentes) possam surgir problemas devido à nova interpretação semântica dos quantificadores, podem-se criar sistemas axiomáticos adequados para eles como extensões de sistemas axiomáticos para cálculos proposicionais. Basta adicionar a um sistema axiomático apropriado deste tipo (ex.: o sistema axiomático de Hilbert e Bernays (1934, 1939)) os axiomas A^* e A^{**}, a condição D e a regra GN (Malinowski, 1993) que se seguem:

(A*)	$\forall x\,(A \to B) \to (A \to \forall x B)$	(Condição: x não ocorre livre em A)
(A**)	$\forall x A(x) \to A(y)$	(Condição: a substituição $x \mapsto y$ é permissível, i.e., y não deve ocorrer ligado onde x ocorre livre em $A(x)$)
(D)	$\exists x A = \neg\forall x\,(\neg A)$, $\forall x A = \neg\exists x\,(\neg A)$	Propriedade de mútua definibilidade dos quantificadores
(GN)	$A(x)/\forall y A(y)$	Regra de generalização

No entanto, as lógicas infinitamente multivalentes suscitam maiores dificuldades. Por exemplo, o conjunto $\{\mathcal{I}\,(A\,(d)) \mid d \in \mathscr{D}\}$ pode não conter um elemento mínimo ou máximo no caso do conjunto $\mathscr{D}$ ser infinito, sendo logo as operações *min* e *max* impossíveis. Um caso problemático é o de qŁ$_{\aleph_0}$, visto que o conjunto de valores de verdade desta lógica não é fechado sob as operações de conjunção e disjunção. Tal como para qŁ$_{\aleph_1}$, demonstrou-se (Scarpelini, 1962) que se os quantificadores $\forall$ e $\exists$ forem introduzidos como no exemplo 2.5.4 qŁ$_{\aleph_0}$ não é axiomatizável.

Partindo da intuição que os quantificadores podem ser tratados como funções sobre o conjunto de pares (x, F), $x \in V$ e F é uma fórmula (ex.: $\forall x F(x) = Q_1(x, F); \exists x F(x) = Q_2(x, F)$), Rosser e Turquette (1952) elaboraram uma teoria geral da quantificação para lógicas finitamente multivalentes. Apresentamos de seguida alguns dos pontos mais importantes desta teoria.

2.5.6. Definição. Um *quantificador generalizado RT*[14] é uma fórmula com a forma

$$Q_i\,(x_1, x_2, ..., x_{m_i}, F_1, F_2, ..., F_{t_i})$$

em que $x_1, x_2, ..., x_{m_i}$ são variáveis nominais e $F_1, F_2, ..., F_{t_i}$ são fórmulas compostas por predicados, variáveis (proposicionais ou nominais) e conetores.

É imediatamente óbvio que um quantificador generalizado RT pode assumir múl-

[14] Iniciais de Rosser e Turquette.

tiplas variáveis e várias fórmulas. A ideia é avaliar as funções Q_i por meio de uma interpretação que atribua a fórmulas valores do conjunto $\{1, 2, ..., n\}$ e em que os quantificadores multivalentes se constroem por meio dos conetores entre as fórmulas. Para tal, Rosser e Turquette propuseram a noção conveniente de *formas normais parciais* (FNPcs) tanto para os conetores como para os quantificadores. No caso dos primeiros, para um dado conetor O (abreviando O_i), a i-ésima FNPc é um *esquema de expressão de fórmula assinalada* (ou seja, uma fórmula com um sinal)[15] $F_i \equiv O(A_1, A_2, ..., A_n)$ contendo apenas fórmulas (assinaladas) $A_1, A_2, ..., A_n$ e em FNC.

2.5.7. Exemplo. Os lados direitos das igualdades são as FNPcs dos conetores $\neg$ e $\vee$ de Ł$_3$:

$$\begin{array}{lcl}
\mathrm{V}[\neg A] & = & \mathrm{F}[A] \\
\mathrm{I}[\neg A] & = & \mathrm{I}[A] \\
\mathrm{F}[\neg A] & = & \mathrm{V}[A] \\
\mathrm{V}[A \vee B] & = & \mathrm{V}[A] \vee \mathrm{V}[B] \\
\mathrm{I}[A \vee B] & = & (\mathrm{I}[A] \vee \mathrm{I}[B]) \wedge (\mathrm{F}[A] \vee \mathrm{I}[A]) \wedge (\mathrm{F}[B] \vee \mathrm{I}[B]) \\
\mathrm{F}[A \vee B] & = & \mathrm{F}[A] \wedge \mathrm{F}[B]
\end{array}$$

Pode-se agora facilmente produzir as FNPcs dos restantes conetores de Ł$_3$ (e as respetivas negações; ex.: $\neg(\mathrm{F}[A \vee B]) = \neg\mathrm{F}[A] \vee \neg\mathrm{F}[B]$), uma vez que é evidente que elas são a enumeração *exaustiva* das interpretações de cada conetor; além disso, são *mutuamente exclusivas* no sentido em que numa qualquer dada interpretação exatamente uma das FNPcs é avaliada como sendo verdadeira.

Dado o esquema da expressão de fórmula assinalada $F_i \equiv O(A_1, A_2, ..., A_n)$, é agora fácil de ver como é que a estratégia das FNPcs se aplica às funções Q_i tal como estas foram definidas acima de modo a especificar o comportamento verofuncional dos quantificadores nas lógicas multivalentes. Em primeiro lugar, recorde-se, da definição 1.1.3, que uma função de verdade para um quantificador Q_i é um mapeamento de conjuntos não vazios de valores de verdade para valores de verdade, i.e., $\widetilde{Q_i} = \left(2^W - \emptyset\right) \longrightarrow W$.

[15] Rosser e Turquette, sem falarem de fórmulas assinaladas (def. 1.1.35), usaram subíndices para indicar o "sinal" (o valor de verdade) de uma fórmula (ex.: p_1 lê-se "p toma o valor 1"). Adotamos aqui a abordagem de Zach (1993) ao método de FNPc de Rosser e Turquette.

2.5.8. Exemplo. As funções de verdade para os quantificadores $\forall$ e $\exists$ de qŁ$_3$ são as seguintes:

$\tilde{\forall}(\{\})$	$=$	$\perp$	$\tilde{\exists}(\{\})$	$=$	$\perp$
$\tilde{\forall}(\{\mathrm{V}\})$	$=$	V	$\tilde{\exists}(\{\mathrm{V}\})$	$=$	V
$\tilde{\forall}(\{\mathrm{V},\mathrm{I}\})$	$=$	I	$\tilde{\exists}(\{\mathrm{V},\mathrm{I}\})$	$=$	V
$\tilde{\forall}(\{\mathrm{V},\mathrm{F}\})$	$=$	F	$\tilde{\exists}(\{\mathrm{V},\mathrm{F}\})$	$=$	V
$\tilde{\forall}(\{\mathrm{V},\mathrm{I},\mathrm{F}\})$	$=$	F	$\tilde{\exists}(\{\mathrm{V},\mathrm{I},\mathrm{F}\})$	$=$	V
$\tilde{\forall}(\{\mathrm{I}\})$	$=$	I	$\tilde{\exists}(\{\mathrm{I}\})$	$=$	I
$\tilde{\forall}(\{\mathrm{I},\mathrm{F}\})$	$=$	F	$\tilde{\exists}(\{\mathrm{I},\mathrm{F}\})$	$=$	I
$\tilde{\forall}(\{\mathrm{F}\})$	$=$	F	$\tilde{\exists}(\{\mathrm{F}\})$	$=$	F

Podemos agora apresentar a seguinte definição:

2.5.9. Definição (Zach, 1993). Para um *esquema* uma fórmula consistindo de quaisquer variáveis de fórmula, variáveis livres e variáveis de termos, um esquema de expressão de fórmula assinalada F é a *i-ésima forma parcial* de $QxA(x)$ sse

(i) Os átomos assinalados em F pertencem a $\{u_{ji}(A(\tau_j)) : 1 \leq j \leq p, 1 \leq i \leq n\} \cup \{w_{ji}(A(\alpha_j)) : 1 \leq j \leq q, 1 \leq i \leq n\}$, em que α_i são variáveis livres e τ_i variáveis de termos.

(ii) Para todo F', F' é uma pré-instância de F,[16] e toda a interpretação $\mathcal{I}$:

(a) Se para todo $d_1, ..., d_q \in \mathscr{D}$ existem $e_1, ..., e_p \in \mathscr{D}$ tais que

$$\mathcal{I} \models F\{e_1/\tau_1, ..., e_p/\tau_p, d_1/\alpha_1, ..., d_q/\alpha_q\}$$

então

$$val_{\mathcal{I}}(QxA'(x)) = v_i.$$

(b) Se para todo $e_1, ..., e_p \in \mathscr{D}$ existem $d_1, ..., d_q \in \mathscr{D}$ tais que

$$\mathcal{I} \not\models F\{e_1/\tau_1, ..., e_p/\tau_p, d_1/\alpha_1, ..., d_q/\alpha_q\}$$

então

$$val_{\mathcal{I}}(QxA'(x)) \neq v_i$$

[16] Dados uma linguagem $\mathcal{L}$ e um conjunto de fórmulas $F \in \mathcal{L}$, uma pré-instância A' de um esquema A é uma fórmula de F contendo ocorrências das variáveis livres e de variáveis de termos de A. Por exemplo, uma pré-instância da fórmula $A(\alpha, \tau)$ é $(P(\alpha, a) \wedge Q(\tau)) \rightarrow P(\tau, \alpha)$.

em que A' é a instância de A determinada por F'.

2.5.10. Proposição. A i-ésima forma normal de $QxA(x)$ é uma FNPc se está em FNC.

2.5.11. Exemplo. As FNPcs dos quantificadores $\forall$ e $\exists$ clássicos são as seguintes:

$$\begin{array}{lcl} \mathrm{V}[\forall xA(x)] & = & \mathrm{V}[A(\alpha)] \\ \mathrm{F}[\forall xA(x)] & = & \mathrm{F}[A(\tau)] \\ \mathrm{V}[\exists xA(x)] & = & \mathrm{V}[A(\tau)] \\ \mathrm{F}[\exists xA(x)] & = & \mathrm{F}[A(\alpha)] \end{array}$$

2.5.12. Exemplo. As FNPcs dos quantificadores $\forall$ e $\exists$ de qŁ$_3$ são as seguintes:

$$\begin{array}{lcl} \mathrm{V}[\forall xA(x)] & = & \mathrm{V}[A(\alpha)] \\ \mathrm{I}[\forall xA(x)] & = & \mathrm{I}[A(\tau)] \wedge (\mathrm{I}[A(\alpha)] \vee \mathrm{V}[A(\alpha)]) \\ \mathrm{F}[\forall xA(x)] & = & \mathrm{F}[A(\tau)] \\ \mathrm{V}[\exists xA(x)] & = & \mathrm{V}[A(\tau)] \\ \mathrm{I}[\exists xA(x)] & = & \mathrm{I}[A(\tau)] \wedge (\mathrm{F}[A(\alpha)] \vee \mathrm{I}[A(\alpha)]) \\ \mathrm{F}[\exists xA(x)] & = & \mathrm{F}[A(\alpha)] \end{array}$$

Finalmente,

2.5.13. Proposição (Rosser & Turquette, 1952). Para uma qualquer fórmula da forma $\forall xF$, suponha-se que F tem apenas ocorrências livres da variável individual z e seja que $F_r(z)$ denota a r-ésima FNPc de F; então, as FNPcs N_r de $\forall xF$ satisfazem as condições-padrão sse a expressão

$$\forall z(F_1(z) \vee ... \vee F_s(z)) \equiv (N_1 \vee ... \vee N_s)$$

em que N_r é a disjunção de conjunções da forma $F_{1,s_1} \wedge F_{2,s_2} \wedge ... \wedge F_{s,s_r}$ e $s_1, ..., s_r$ é uma sequência de valores de verdade tal que $val(\phi) = r$ para uma qualquer valoração $val(F_i) = i_r$, é demonstrável no cálculo de predicados bivalente.

Obviamente, o conjunto de todas as N_r de uma qualquer fórmula equivale à sua tabela de verdade. Por "satisfazer as condições-padrão" entende-se que $\forall xF$ toma

um valor designado sse F toma sempre um valor designado (cf. prop. 2.2.1). Esta definição mais lata de condições-padrão permite a axiomatização dos cálculos de predicados n-valentes pela adição a A1-A7 dos seguintes esquemas A8-A10 e da regra de generalização GGN:

(A8)	$\forall x A(x) \to A(y)$	(Condição: a substituição $x \mapsto y$ é permissível, i.e., y não deve ocorrer ligado onde x ocorre livre em $A(x)$)
(A9)	$\forall x\,(A_1 \to A_2) \to (A_1 \to \forall x A_2)$	(Suposição: x não é livre em A_1)
(A10)	$N_r\,(Q_i\,(x_1, ..., x_{m_i}, A_1, ...A_{t_i})) \to$ $J_r\,(Q_i\,(x_1, ..., x_{m_i}, A_1, ...A_{t_i}))$	em que $1 \leq r \leq n$, $1 \leq i \leq c$
(GGN)	$A/\forall x A$	

2.5.2 Quantificação nas lógicas difusas

Com respeito à quantificação para as lógicas difusas de norma-t, esta é matematicamente complexa e frequentemente resulta em sistemas não axiomatizáveis, como já se referiu no caso de $\mathrm{qŁ}_{\aleph_1}$. Com efeito, de modo a obter-se $\mathrm{qL}\,(\star)$ adiciona-se a $\mathrm{L}\,(\star)$ um domínio de quantificação $\mathscr{D}$ e define-se $\forall$ e $\exists$ como as operações *inf* e *sup* (exemplo 2.5.4). Obtemos qLB adicionando aos axiomas e regras LB (def. 2.4.33), para C denotando uma qualquer fórmula fechada (i.e., uma fórmula sem variáveis livres) e em que x não é livre, os seguintes axiomas:

(qLB1)	$\vdash \forall x A \to A_x(d)$
(qLB2)	$\vdash A_x(d) \to \exists x A$
(qLB3)	$\vdash \forall x\,(C \to A) \to (C \to \forall x A)$
(qLB4)	$\vdash \forall x\,(A \to C) \to (\exists x A \to C)$
(qLB5)	$\vdash \forall x\,(C \vee A) \to (C \vee \forall x A)$
(qLB6)	Se $\vdash A_x(d)$ então $\vdash \forall x A$

Porém, apenas qLG possui hoje uma axiomatização adequada, pela adição a LB1-LB8 e qLB1-qLB6 do axioma da proposição 2.4.37 juntamente com a regra MP (Hájek, 1997). No que respeita a qŁ$_\aleph$, existe uma generalização quantificada de um sistema ao estilo de Pavelka[17] para a lógica difusa proposicional de Łukasiewicz (Novák, 1990; Novák et al., 1999). Não há presentemente nenhum sistema axiomático adequado para qLΠ. Tal como no caso dos seus sistemas proposicionais, qLB, qŁ$_\aleph$, qLG e qLΠ encontram uma semântica algébrica apropriada nas álgebras LB, MV, G e Π, respetivamente (Esteva et al., 2003).

Assim sendo, a especificação de uma interpretação para lógica difusa de primeira ordem (LDPO) é especialmente importante.

2.5.14. Definição. Um triplo $(\mathscr{D}, \Theta, \delta)$ é uma interpretação $\mathcal{I}$ para LDPO se

- $\mathscr{D} \neq \emptyset$, $\mathscr{D}$ é um domínio.
- Θ é um mapeamento $\mathscr{D}^n \longrightarrow [0,1]$ tal que para cada predicado P de aridade $n > 0$ temos $val_{\mathcal{I}}(P(x_1, ..., x_n)) \in [0,1]$.
- δ é uma atribuição de um membro de $\mathscr{D}$ a cada constante individual d tal que $\mathcal{I}(d) \in \mathscr{D}$.

2.5.15. Exemplo. As condições de verdade para qŁ$_\aleph$ são aquelas da definição 2.4.7 e do exemplo 2.5.4.

2.5.16. Exemplo. Seja dado o domínio $\mathscr{D} = \{C, S, B, F, W\}$ em que as letras denotam animais; podemos definir uma *relação* $MS(\mathscr{D} \times \mathscr{D})$ "o animal x é muito mais pequeno do que o animal y" do seguinte modo:

MS	C	S	B	F	W
C	0	0.5	0.8	1	1
S	0	0	0.7	1	1
B	0	0	0	0.9	1
F	0	0	0	0	0.5
W	0	0	0	0	0

[17] Num sistema ao estilo de Pavelka (Pavelka, 1979), além das fórmulas ϕ considera-se ainda o seu grau de pertença a um conjunto difuso $\Sigma^\sim$, sendo que os graus de pertença $\Sigma^\sim(\phi)$ são de facto valores de verdade formando um reticulado residuado $\mathscr{L} = (\mathscr{A}, \cap, \cup, \star, \rightarrow, 0, 1)$. Bergmann (2008) dá vários exemplos de abordagens deste tipo às lógicas difusas de Łukasiewicz tanto proposicionais como de primeira ordem.

Obviamente, MS é uma relação *difusa* na medida em que $MS : \mathscr{D}^2 \longrightarrow [0,1]$. Seja agora que $Ms(x,y)$ é o *predicado binário difuso* que corresponde à relação MS, ou seja, $Ms : \mathscr{D}^2 \longrightarrow [0,1]$; seja ainda que MS estabelece o menor valor de verdade possível do predicado $Ms(x,y)$. Temos então que, por exemplo e abreviando $val_{\mathcal{I}}(A)$ para uma qualquer fórmula A como $/A/$, $/\forall x \forall y ((C(x) \wedge S(y)) \longrightarrow Ms(x,y))/ = 1$ em qŁ$_{\aleph}$. De facto, pelo exemplo 2.5.3, $/\forall x (C(x))/ = 0$ e $/\forall y (S(y))/ = 0$, ou seja, o grau de verdade em que um qualquer x ou y pode ser membro dos conjuntos C e S respetivamente é o *minimum* dos graus de pertença no intervalo $[0,1]$ de um qualquer objeto a um qualquer conjunto, i.e., 0. Por seu lado, a valoração da conjunção $C(x) \wedge S(y)$ é também ela 0 e por fim $/(C(x) \wedge S(y)) \longrightarrow Ms(x,y)/ = 1$, dado que 0.5 é o grau de verdade mínimo para o predicado $Ms(x,y)$ quando $x \in C$ e $y \in S$, ou seja, temos que o valor de verdade do antecendente é menor do que o do consequente.

2.5.17. RESULTADO. Os resultados 2.4.43-5 servem *mutatis mutandis* para as versões quantificadas destas lógicas difusas de norma-t. Adicionalmente, os seus quantificadores são normais.

2.6 Exercícios

1. Construa as tabelas de verdade das fórmulas seguintes:

(a) $A \rightarrow_{L3} A$; (b) $A \rightarrow_{BI} (\neg_{BI} A \wedge_{BI} B)$; (c) $A \rightarrow_{L3} \neg_{L3} A$; (d) $\phi \rightarrow_{KS} \psi$;
(e) $(\phi \rightarrow_{BE} \phi) \rightarrow_{BE} \psi$; (f) $P \rightarrow_{KW} \neg_{KW} (P \vee_{KW} Q)$;
(g) $(P \leftrightarrow_{L3} Q) \wedge_{L3} (P \rightarrow_{L3} \neg_{L3} Q)$

2. Verifique se as fórmulas acima têm fórmulas equivalentes nos restantes sistemas lógicos multivalentes da Secção 2.4.

3. Determine se as tautologias clássicas T1 - T6 (resultado 1.1.21) o são em Ł$_3$, B$_3^I$, B$_3^E$, K$_3^S$ e K$_3^W$.

4. Determine se as contradições clássicas da proposição 1.1.23 o são nestes sistemas.

5. Dada uma interpretação $\mathcal{I}$ e uma valoração $val_{\mathcal{I}}(\phi)$ (abreviada como $/\phi/$) para uma qualquer fórmula ϕ, seja que

$/P/ = 1$
$/Q/ = 0.8$
$/R/ = 0.5$
$/S/ = 0.2$
$/T/ = 0$

Determine a valoração de cada uma das seguintes fórmulas na lógica difusa $Ł_{\aleph}$:

(a) $P \wedge Q$; (b) $P \wedge T$; (c) $P \rightarrow Q$; (d) $T \rightarrow P$; (e) $R \rightarrow S$;
(f) $(R \wedge S) \rightarrow R$; (g) $P \wedge \neg P$; (h) $P \vee \neg P$; (i) $S \rightarrow (T \vee \neg T)$;
(j) $(S \wedge \neg S) \rightarrow R$; (k) $(P \leftrightarrow \neg Q) \wedge (\neg P \leftrightarrow Q)$;
(l) $P \,\&\, R$; (m) $P \,\&\, \neg P$

6. Indique se os seguintes argumentos são válidos nas lógicas difusas estudadas acima:

(a) $P \rightarrow \neg P / \neg P$
(b) $P \vee P / P$

7. Mostre de modo informal que a lógica de Belnap B_4 não é uma extensão da lógica clássica.

8. Conceba uma lógica multivalente e indique informalmente os aspetos que se encontram na base da sua conceção.

3 Satisfazibilidade e automatização do raciocínio: O cálculo de resolução

3.1 Demonstração automática de teoremas e o problema SAT

3.1.1 Demonstração de teoremas e o problema da decisão

Dada uma teoria lógica, i.e., um conjunto de fórmulas numa linguagem $\mathcal{L}^X$, somos frequentemente confrontados com a necessidade de demonstrar que uma certa fórmula ϕ em $\mathcal{L}^X$ é verdadeira – sempre (validade) ou em alguma interpretação (satisfazibilidade). Por outras palavras, perguntamos se $\phi \in \mathcal{L}^X$. Este é aquilo que chamamos o *problema da decisão*, sendo que se diz de uma teoria que é ela é *decidível* se existe um *procedimento efetivo de decisão* ou, o que é o mesmo, um *algoritmo* que resolve o problema da decisão. Por "resolver o problema da decisão" entende-se que o algoritmo (i) pára e (ii) responde "Sim" se $\phi \in \mathcal{L}^X$ ou "Não" se $\phi \notin \mathcal{L}^X$. Uma teoria lógica é *indecidível* se não é decidível. Se além de (i) e (ii) o algoritmo (iii) não pára se $\phi \notin \mathcal{L}^X$, então diz-se que $\mathcal{L}^X$ é *semidecidível*: sabemos com certeza apenas que o algoritmo pára se $\phi \in \mathcal{L}^X$; caso contrário, o algoritmo pode parar ou não.

Recorde-se o teorema da dedução (1.2.8-9): este teorema expressa o facto que a validade e a satisfazibilidade são conceitos duais no sentido em que uma qualquer fórmula ϕ é válida sse $\neg\phi$ é insatisfazível. Isto implica, como se verá abaixo, o resultado fundamental que o problema da decisão pode ser reduzido ao problema da satisfazibilidade (ou problema SAT).

As tabelas de verdade são procedimentos efetivos de decisão em $\mathcal{L}^{Prop}$ apenas para fórmulas com um número n de variáveis proposicionais relativamente pequeno, deixando de o ser para um maior n de variáveis uma vez que o número de linhas de uma tabela de verdade clássica é determinado por 2^n. Além disso, independentemente

de n, este procedimento não está disponível para a LPO clássica. Aplicam-se outros métodos para LPO, mas apenas fragmentos desta são decidíveis, ou seja, a LPO é indecidível. Este resultado é conhecido como o *teorema de Church-Turing* (Church, 1936a, 1936b; Turing, 1936-7).

Assim sendo, precisamos tanto de procedimentos de decisão mais eficientes para LP como de procedimentos de redução de uma asserção qualquer em LPO a uma asserção equisatisfazível, ou equivalente em termos de satisfazibilidade, em LP. Estes métodos existem já, mas questões de complexidade podem torná-los inúteis para computadores humanos, a menos que possam ser implementados em máquinas que conseguem lidar com altos níveis de complexidade computacional e retornar uma resposta sim-ou-não em tempo útil. Falamos aqui de *automatização do raciocínio*, nomeadamente no que respeita à demonstração de teoremas, um campo que mantém muita gente ocupada há já algumas décadas.

Como se referiu na Introdução, as lógicas multivalentes são hoje uma componente essencial da lógica matemática e da ciência dos computadores, razão pela qual a automatização da demonstração de teoremas é um problema premente nestas lógicas. Felizmente, há procedimentos que nos permitem reduzir a demonstração de teoremas em lógicas multivalentes a demonstrações clássicas por meio do problema SAT. Nos parágrafos que se seguem começamos por discutir a automatização da demonstração de teoremas baseada na satisfazibilidade em LC por meio do cálculo de resolução; no capítulo seguinte estendemos a resolução às lógicas multivalentes.

3.1.2 Algumas notas históricas acerca da demonstração automática de teoremas

A ambição de automatizar a dedução – ou seja, encontrar métodos algorítmicos para decidir acerca da verdade lógica de proposições – não é recente; é anterior em relação não só ao campo da inteligência artificial (IA), o qual emergiu nos anos 1950, mas ainda à formalização da lógica com Boole e Frege nos finais do séc. XIX. De facto, pode dizer-se que esta ambição data já do séc. XVII, quando Leibniz imaginou a possibilidade de um *calculus ratiocinator*, um "cálculo do raciocínio" capaz de reduzir a solução de um qualquer problema adequadamente formalizado a um procedimento mecânico; esta formalização adequada seria realizada por meio daquilo que ele chamou a *lingua universalis*, uma linguagem conceptual universal que teria por interpretação uma *characteristica universalis* ou teoria geral dos signos.

Desde os inícios do séc. XX, o programa de formalização da matemática proposto por D. Hilbert e a sua explicitação do *Entscheidungsproblem* ou problema da decisão (e.g., Hilbert, 1900; Hilbert & Ackermann, 1928) haviam motivado uma definição

matemática clara de processo algorítmico. Apesar de terem resolvido o problema de forma negativa (i.e., não há nenhum algoritmo que decida se uma proposição em LPO é demonstrável em LPO),[1] A. Church e A. Turing deram, independentemente um do outro, a definição requirida de algoritmo ou função efetivamente calculável (Church, 1936a, 1936b; Turing, 1936-7). Embora as duas sejam equivalentes, foi a solução de Turing, em que protagonizavam as famosas máquinas de Turing, que ofereceu uma definição mais clara e intuitiva de procedimento algorítmico.[2]

Uma **máquina de Turing** (MT) é um conjunto finito de n-tuplos — para n habitualmente igual a 4, 5 ou 6 — descrevendo, com mais ou menos elementos n, um estado da máquina e a prescrição de uma ação e da transição para o estado seguinte. Este n-tuplo é de facto um programa finito capaz de manipular uma fita (infinita) utilizando um dispositivo de leitura e escrita (DLE), pelo que se pode chamar o controlo finito. Consideramos aqui um 5-tuplo $\mathcal{M} = (Q, \Sigma, \Gamma, s, \delta)$ em que

- $Q = \{q_0, q_1, ..., q_n\}$ para q_i designando um estado da máquina; os dois estados de paragem (*halt*) h_a e h_r não são elementos de Q.
- Σ é o alfabeto (finito) de *input* e Γ o alfabeto (finito) da fita, com $\Gamma = \{0, 1, \mathrm{X}\}$ comummente e $\Sigma \subseteq \Gamma$.
- $s = q_0$.
- δ é a função de transição

$$\delta : Q \times (\Gamma \cup \{\Delta\}) \longrightarrow (Q \cup \{h_a, h_r\} \times (\Gamma \cup \{\Delta\}) \times \{D, E, S\})$$

em que Δ designa o símbolo vazio (habitualmente: 0), h_a e h_r designam paragens de aceitação e rejeição, respetivamente, e D, E e S designam as direções "direita", "esquerda" e "estacionário", respetivamente, para o DLE.

Vejamos um exemplo concreto: a fórmula

$$\delta\,(q_0, 0) = (q_1, 1, D)$$

tem a interpretação "se $\mathcal{M}$ está no estado q_0 e o símbolo na fita é 0, então $\mathcal{M}$ substitui 0 por 1, move o DLE uma célula para a direita e entra no estado q_1".
O programa de uma MT é simplesmente a lista de todas as suas funções de transição.

[1] Este é conhecido como o *teorema* de Church-Turing.
[2] Com efeito, a *tese* de Church-Turing enuncia que a classe das funções efetivamente calculáveis é precisamente a classe das funções computáveis por uma máquina de Turing.

Duas noções são fundamentais no comportamento de uma MT:

1. Dizemos que uma MT $\mathcal{M}$ com alfabeto de *input* Σ ***aceita*** uma linguagem $\mathcal{L} \subseteq \Sigma^*$ se $\mathcal{L}(\mathcal{M}) = \mathcal{M}$ em que

$$\mathcal{L}(\mathcal{M}) = \{x \in \Sigma^* \mid x \text{ é aceite por } \mathcal{M}\},$$

   ou seja, se, dado um *input* x, $\mathcal{M}$ acaba por parar num estado de aceitação h_a. Mais formalmente, se $\mathcal{M} = (Q, \Sigma, \Gamma, s, \delta)$ é uma MT e $x \in \Sigma^*$, x é aceite por $\mathcal{M}$ se

$$q_0 \Delta x \vdash^*_{\mathcal{M}} w h_a y$$

   para sequências $w, y \in (\Gamma \cup \{\Delta\})^*$, ou seja, se começando na configuração inicial correspondente ao *input* x, $\mathcal{M}$ acaba por parar no estado de aceitação (logo, não rejeitando ou não ficando "pendurada" numa sequência $x \notin \mathcal{L}$).

2. $\mathcal{M}$ ***decide*** $\mathcal{L}$ se $\mathcal{M}$ computa a função característica $\chi_{\mathcal{L}} : \Sigma^* \longrightarrow \{0, 1\}$, ou seja, $\mathcal{M}$ decide se $x \in \mathcal{L}$ de modo a determinar o valor de $\chi_{\mathcal{L}}(x)$ por

$$\chi_{\mathcal{L}}(x) = \begin{cases} 1 & \text{se } x \in \mathcal{L} \\ 0 & \text{caso contrário} \end{cases}.$$

De 1 e 2 segue-se o importante resultado que uma linguagem $\mathcal{L}$ é *recursivamente enumerável* se há uma MT que aceita $\mathcal{L}$ e $\mathcal{L}$ é *recursiva* se há uma MT que decide $\mathcal{L}$.

Estava aberto o caminho para os primeiros passos na automatização da dedução e estes foram dados nos anos 1950, com a programação em 1954 por M. Davis de um computador com o procedimento de Presburger (Davis, 1957) e com a *Logic Theory Machine* de A. Newell e colaboradores (Newell et al., 1957), a qual se pode considerar o primeiro programa de facto de IA. Este programa emulava o raciocínio dos *Principia mathematica* de Whitehead e Russell (1910) e era sobretudo heurístico; aquele, embora essencialmente algorítmico, não foi além de demonstrar que a soma de dois números pares é um número par, sobretudo porque o procedimento de Presburger tem complexidade mais do que exponencial. Por esta altura foram publicados vários resultados importantes em procedimentos de demonstração e demonstrações simplificadas de completude para LPO, todos com impacto na conceção dos primeiros demonstradores de teoremas (ex.: Beth, 1955; Hintikka, 1955; Quine,

1955; Schütte, 1956).

Na verdade, os utensílios básicos para a automatização da dedução em LPO já tinham sido encontrados nos anos 1920, nomeadamente por Skolem (e.g., 1920) e Herbrand (1930) (ver abaixo), e o verdadeiro progresso deu-se nos finais dos anos 1950 com Davis e Putnam (1958; 1960). Estes usaram FNCs para testar a insatisfazibilidade, fazendo uso de quatro regras, a saber, a regra da tautologia, a regra do literal único (*one-literal rule*), a regra do literal puro (*pure-literal rule*) e a regra da separação (*splitting rule*). O seu era mais eficiente do que o método da multiplicação de Gilmore (1960), o qual também se baseava no teorema de Herbrand (versão II; ver abaixo) que, por sua vez, sugeria um procedimento de refutação. Este método foi melhorado por Davis, Logemann e Loveland (1962) com vista à testagem da satisfazibilidade no que é agora conhecido como o procedimento de Davis-Putnam-Logemann-Loveland (DPLL). Pode dizer-se que uma "revolução" teve lugar quando A. J. Robinson (1965) redescobriu a regra de resolução, primeiramente descoberta por A. Blake (1937), e a unificação, primeiro descoberta por Prawitz (1960). O grupo do *Argonne National Laboratory* dirigido por L. Wos e G. Robinson melhorou a eficácia da resolução; com efeito, era necessário reduzir o espaço de procura dos resolventes gerados. De modo a atingir este objetivo, introduziram a fatorização (*factoring*), a preferência por unidades (*unit preference*) e ainda a estratégia do conjunto de apoio (*set of support*); mais tarde introduziram a igualdade, um trabalho que resultou nas técnicas da modulação e da paramodulação. Todas estas técnicas foram implementadas no demonstrador automático Otter (ver Kalman, 2001), cujo sucessor atual é o Prover9/Mace4.

3.1.3 O problema SAT e a demonstração de teoremas

Os desenvolvimentos acima estão de um modo ou de outro ligados ao problema SAT, nomeadamente à procura dos chamados *SAT solvers*, ou seja, algoritmos para resolver (instâncias d)o problema SAT.

3.1.1. Definição (problema booleano da satisfazibilidade, ou problema SAT). Dada uma fórmula $A(x_1, ..., x_n)$, pergunta-se se A pode receber o valor V em alguma atribuição de valores de verdade V ou F às variáveis x_i, $1 \leq i \leq n$. Diz-se que uma fórmula (proposicional) $A(x_1, ..., x_n)$ é *satisfazível* se valores de verdade podem ser atribuídos às suas variáveis x_i de modo a fazer A verdadeira.

Do ponto de vista da teoria da complexidade, SAT é um problema NP-completo

(com efeito, foi o primeiro problema do qual se demonstrou ser NP-completo; Cook, 1971).

Damos aqui uma introdução informal à problemática das classes de complexidade dos problemas de decisão.

Seja a complexidade de um problema de decisão definida em termos de tempo, ou seja, *complexidade temporal*; então a complexidade temporal de um problema de decisão é o tempo que uma máquina de Turing leva até resolver o problema como uma função do tamanho n do *input*, i.e., $T(n)$. Dizemos que o problema é *polinomial* se for resolvido num tempo polinomial, ou seja, uma quantidade temporal limitada acima por uma função polinomial do tamanho das instâncias ou casos particulares do problema; formalmente, $T(n) = O\left(n^k\right)$ para uma qualquer constante k e $O(\cdot)$ uma notação (*grande O*) que caracteriza funções de acordo com taxas de crescimento (assintótico).

Se o problema é resolúvel em tempo polinomial por uma máquina de Turing *determinística*, ou seja, uma máquina de Turing que prescreve uma única acção para cada estado, o problema pertence à classe de complexidade **P**. Dizemos então que existe um algoritmo polinomial para o problema. Se o problema é resolúvel em tempo polinomial mas por uma máquina de Turing *não determinística*, ou seja, uma máquina de Turing que prescreve várias acções para cada estado, então o problema pertence à classe de complexidade **NP** (abreviatura de ***n****ondeterministic* ***p****olynomial*). Os problemas desta classe são aqueles para os quais, se a resposta for "sim", então há uma prova que pode ser verificada em P (por exemplo, se uma dada fórmula é satisfazível; cf. problema SAT); por outro lado, se a resposta for "não", então qualquer prova de que a resposta é "sim" deve ser declarada inválida.

Temos pois que P é a classe de complexidade de problemas de decisão "tratáveis", enquanto a classe NP é aquela dos problemas "razoáveis". Ainda por outras palavras, um problema P pode ser *resolvido* rapidamente, enquanto um problema NP pode ser *verificado* rapidamente.

Diz-se que um problema de decisão pertence à classe **NPC** (abreviando "NP-completo") se não se conhece nenhum método eficiente para identificar a solução em primeiro lugar, embora esta se possa verificar rapidamente. Ou seja, estes problemas são mais difíceis de resolver do que de verificar, não podendo em princípio ser resolvidos em tempo polinomial.

Note-se que no âmbito daquele que é considerado um (o?) problema maior da ciência dos computadores, a saber, $\mathrm{P} \stackrel{?}{=} \mathrm{NP}$, se $\mathrm{P} = \mathrm{NP}$ temos então que $\mathrm{P} = \mathrm{NP} = \mathrm{NPC}$; caso contrário, $\mathrm{P} \subset \mathrm{NP}$ e $\mathrm{NPC} \subset \mathrm{NP}$, mas $\mathrm{NP} \cap \mathrm{NPC} = \emptyset$. (Ver, por ex., Fortnow, 2009, para uma introdução matematicamente acessível a este problema.)

Em termos gerais, na demonstração de teoremas interessa-nos saber se um teorema é válido. De modo a obter este resultado, um problema é transformado numa fórmula ou num conjunto de fórmulas numa linguagem lógica proposicional ou de primeira ordem; a fórmula é então introduzida como *input*, tal como está ou após passar por um processo de transformação ou tradução, e é sujeita a uma manipulação por um método de demonstração do qual se espera que produza um *output*, preferencialmente em tempo útil. Os métodos de demonstração baseados no SAT operam sobre fórmulas verificando se são satisfazíveis ou insatisfazíveis. Neste último caso temos um método de refutação, ou seja, um método que testa a insatisfazibilidade de um conjunto de fórmulas: se um conjunto de fórmulas exprimindo um teorema negado é insatisfazível, então o teorema é válido e pode-se produzir uma demonstração. No caso da testagem da satisfazibilidade, esta tem em vista a produção de um modelo no caso de se verificar que uma fórmula é satisfazível. A resolução e o procedimento DPLL são os exemplos principais destes dois tipos de teste e têm em comum o facto que operam sobre fórmulas em FNC. A testagem pela (in)satisfazibilidade mostra-se particularmente importante na LPO visto que as fórmulas (em FNC) têm de passar pelo processo de skolemização (ver abaixo): este processo não preserva a validade, mas preserva a satisfazibilidade de uma fórmula.

3.2 As formas normais de Skolem e o teorema de Herbrand

No Capítulo 1 introduzimos a lógica clausal ou de cláusulas. De modo a introduzir aspetos gerais da demonstração por resolução precisamos ainda dos importantes resultados de Skolem e Herbrand que mencionámos acima. Com efeito, uma vez obtida uma fórmula em FNP, os próximos passos antes de aplicar a resolução são (1) a transformação da matriz numa FNC e (2) a remoção dos quantificadores existenciais. O passo (1) foi abordado acima; o passo (2) é aquilo que chamamos *skolemização* e que consiste no procedimento seguinte:

3.2.1. Proposição. Seja que uma fórmula F está na FNP $(Q_1 x_1) \ldots (Q_n x_n)\, M$ em que M está em FNC. Seja Q_r, $1 \leq r \leq n$, um quantificador existencial no prefixo

de F.

(i) Se não há nenhum quantificador universal antes de Q_r escolhemos uma nova constante c que não ocorre em M, substituímos todas as x_r em M por c e removemos $Q_r x_r$ do prefixo.

(ii) Se antes de Q_r há os quantificadores universais $Q_{s_1}, ..., Q_{s_m}$, i.e., $1 \leq s_1 < s_2 < ... < s_m < r$, selecionamos um símbolo novo f de função de aridade m não ocorrendo em M, substituímos todos os Q_r em M por $f(x_{s_1}, ..., x_{s_m})$ e removemos $Q_r x_r$ do prefixo.

3.2.2. Exemplo. Partindo da fórmula

$$(*) \qquad (\exists x)(\forall y)(\forall z)(\forall u)(\exists v)((P(x) \vee Q(y)) \wedge R(z,u,v)),$$

pela aplicação do procedimento da proposição 3.2.1 obtemos

$$(**) \qquad (\forall y)(\forall z)(\forall u)((P(a) \vee Q(y)) \wedge R(z,u,f(y,z,u)))$$

3.2.3. Definição. $(**)$ está na *forma normal de Skolem* (FNS). A constante a e a função $f(y,z,u)$ que substituem as variáveis existenciais de $(*)$ chamam-se *constante de Skolem* e *função de Skolem*, respetivamente.

Como se sabe, os quantificadores universais podem ser removidos de $(**)$. De modo a prosseguir com a resolução, uma fórmula em FNS tem de se representar como um conjunto de cláusulas. Por exemplo, $(**)$ deve representar-se pelo conjunto

$$\{P(a) \vee Q(y), R(z,u,f(y,z,u))\}.$$

É importante frisar que o procedimento de skolemização significa que $(*)$ e $(**)$ *não* são logicamente equivalentes; contudo, para os fins da testagem em SAT, temos que elas são equisatisfazíveis ou equivalentes em termos de satisfazibilidade, i.e.,

$$(*) \equiv_{sat} (**).$$

3.2.4. Definição (equisatisfazibilidade). Duas fórmulas ϕ e ψ são *equivalentes em termos de satisfazibilidade* ou *equisatisfazíveis*, o que se denota por $\phi \equiv_{sat} \psi$, sse

são ambas satisfazíveis ou são ambas insatisfazíveis.

3.2.5. TEOREMA. Seja $\mathcal{C}$ um conjunto de cláusulas representando a FNS de uma fórmula ϕ. Então, ϕ é insatisfazível sse $\mathcal{C}$ é insatisfazível.

Demonstração. (ex.: Chang & Lee, 1973, p. 48-9.) □

Este último teorema exige que se especifique em que condições $\mathcal{C}$ é insatisfazível. Tal especificação foi levada a cabo por Herbrand:

3.2.6. TEOREMA (Herbrand, 1930 - versão II).[3] Um conjunto $\mathcal{C}$ de cláusulas é insatisfazível sse existe uma conjunto finito insatisfazível $\mathcal{C}'$ de instâncias básicas (*ground*, em inglês) de cláusulas de $\mathcal{C}$.

Embora não "perfeitos", os resultados de Herbrand são fundamentais em mais do que um sentido. Para começar, este teorema diz-nos que de modo a verificar se $\mathcal{C}$ é insatisfazível, precisamos apenas de considerar as interpretações de Herbrand, pois um conjunto de cláusulas $\mathcal{C}$ é insatisfazível sse $\mathcal{C}$ é falso em todas as interpretações de Herbrand. Por seu lado, isto quer dizer que temos apenas de considerar o universo de Herbrand (vs. todos os domínios possíveis). De seguida introduzimos a terminologia necessária para compreender isto.

Seja $\mathcal{C}$ um conjunto de cláusulas; então as seguintes definições valem:

3.2.7. DEFINIÇÃO. O *universo de Herbrand* de $\mathcal{C}$, denotado por $H_{\mathcal{C}}$, é o conjunto de todos os *termos básicos* construídos a partir das constantes e funções de $\mathcal{C}$ do modo seguinte:

(i) $\mathcal{C}$ não contém nenhuma função; então, $H_{\mathcal{C}}$ é o conjunto das constantes que ocorrem em $\mathcal{C}$;

(ii) se não ocorrem quaisquer funções ou constantes em $\mathcal{C}$, então $H_{\mathcal{C}}$ consiste numa única constante, por exemplo, $H_{\mathcal{C}} = \{a\}$;

(iii) se $\mathcal{C}$ contém uma função, então $H_{\mathcal{C}}$ é infinito e H_i, $1 \leq i \leq n$ para n infinito é o *conjunto de constantes de nível i* de $\mathcal{C}$.

[3] Decidimos apresentar em primeiro lugar aquela que é conhecida como a versão II deste teorema; demonstramo-la após a enunciação e demonstração da versão I do mesmo teorema.

3.2.8. EXEMPLO. (i) Seja $\mathcal{C} = \{P(b), \neg P(x) \vee Q(y)\}$. Então, $H_{\mathcal{C}} = \{b\}$.

(ii) Seja $\mathcal{C} = \{P(x) \vee Q(x), R(z), T(y) \vee \neg S(y)\}$. Então faz-se $H_{\mathcal{C}} = \{a\}$.

(iii) Seja $\mathcal{C} = \{P(f(x)), a, g(y), b\}$. Então,

$$
\begin{aligned}
H_0 &= \{a, b\} \\
H_1 &= \{a, b, f(a), f(b), g(a), g(b)\} \\
H_2 &= \{a, b, f(a), f(b), g(a), g(b), f(f(a)), f(f(b)), f(g(a)), f(g(b)), \\
&\quad g(f(a)), g(f(b)), g(g(a)), g(g(b))\} \\
&\vdots \\
H_{\mathcal{C}} &= \{a, b, f(a), f(b), g(a), g(b), f(f(a)), f(f(b)), f(g(a)), f(g(b)), \\
&\quad g(f(a)), g(f(b)), g(g(a)), g(g(b)), ...\}
\end{aligned}
$$

3.2.9. DEFINIÇÃO. Uma *instância básica* de uma cláusula C de $\mathcal{C}$ é uma cláusula obtida pela substituição de variáveis em C por membros de $H_{\mathcal{C}}$. Uma *instância de Herbrand* de C é uma instância básica $C\theta$ de C tal que θ se baseia em $\mathcal{C}$.

3.2.10. DEFINIÇÃO. A *base de Herbrand* de $\mathcal{C}$, denotada por $H(\mathcal{C})$, é o conjunto de todas as instâncias de Herbrand de átomos que ocorrem nas cláusulas de $\mathcal{C}$.

3.2.11. DEFINIÇÃO. Uma *interpretação de Herbrand (interpretação H)* para $\mathcal{C}$, denotada por $H\mathcal{I}_{\mathcal{C}}$, é um subconjunto de $H(\mathcal{C})$ tal que o valor de verdade V é atribuído a todos os elementos de $H\mathcal{I}_{\mathcal{C}}$ e o valor de verdade F é atribuído a todos os átomos em $H(\mathcal{C}) - H\mathcal{I}_{\mathcal{C}}$. Se uma interpretação $H\mathcal{I}_{\mathcal{C}}$ faz com que todas as cláusulas de $\mathcal{C}$ sejam verdadeiras, então $H\mathcal{I}_{\mathcal{C}}$ é um *modelo de Herbrand* para $\mathcal{C}$ e diz-se de $\mathcal{C}$ que é *H-satisfazível*; caso $\mathcal{C}$ não tenha um modelo de Herbrand, então $\mathcal{C}$ é *H-insatisfazível*. De modo geral, dados n elementos em $H(\mathcal{C})$, há 2^n interpretações H para $\mathcal{C}$.

3.2.12. EXEMPLO. Seja que $\mathcal{C} = \{P(x) \vee Q(x), R(f(y))\}$. Então tem-se que $H_{\mathcal{C}} = \{a, f(a), f(f(a)), ...\}$ e $H(\mathcal{C}) = \{P(a), Q(a), R(a), P(f(a)), Q(f(a)), R(f(a)), ...\}$. As seguintes são interpretações H para $\mathcal{C}$:

$$
\begin{aligned}
H\mathcal{I}_{\mathcal{C}_1} &= \{P(a), Q(a), R(a), P(f(a)), Q(f(a)), R(f(a)), ...\} \\
H\mathcal{I}_{\mathcal{C}_2} &= \{\neg P(a), \neg Q(a), \neg R(a), \neg P(f(a)), \neg Q(f(a)), \neg R(f(a)), ...\} \\
H\mathcal{I}_{\mathcal{C}_3} &= \{P(a), Q(a), \neg R(a), P(f(a)), Q(f(a)), \neg R(f(a)), ...\}
\end{aligned}
$$

É fácil de ver que $\mathcal{C}$ é satisfeito por $H\mathcal{I}_{\mathcal{C}_1}$, mas é falsificado por $H\mathcal{I}_{\mathcal{C}_2}$ e $H\mathcal{I}_{\mathcal{C}_3}$. Logo, $H\mathcal{I}_{\mathcal{C}_1}$é um modelo de Herbrand de $\mathcal{C}$ e $\mathcal{C}$ é H-satisfazível.

3.2.13. TEOREMA. Um conjunto $\mathcal{C}$ de cláusulas é insatisfazível sse $\mathcal{C}$ é falso em todas as interpretações H.

Demonstração. ($\Rightarrow$) Trivial, uma vez que um qualquer conjunto de cláusulas $\mathcal{C}$ é insatisfazível sse é falso em todas as interpretações de um domínio qualquer.

($\Leftarrow$) Suponha-se que $\mathcal{C}$ é falso em todas as interpretações H. Seja que $\mathcal{C}$ é satisfazível; então existe uma interpretação $\mathcal{I}$ para um qualquer domínio $\mathscr{D}$ tal que $\mathcal{C}$ é verdadeiro em $\mathcal{I}$. Seja agora que $\mathcal{I}^*$ é uma interpretação H correspondente a $\mathcal{I}$. Então $\mathcal{C}$ é verdadeiro numa interpretação H. Mas isto vai contra a suposição de que $\mathcal{C}$ é falso em todas as interpretações H, pelo que $\mathcal{C}$ deve ser insatisfazível. □

Aqui é, porém, onde os resultados de Herbrand se podem tornar problemáticos, uma vez que podem existir infinitamente muitas interpretações H. Ora, acontece que o primeiro requisito de um algoritmo é que o número de passos que o constituem seja finito. Com vista à organização de todas as interpretações H de modo sistemático podemos aplicar a noção de árvore semântica.

3.2.14. DEFINIÇÃO. (1) Uma *árvore semântica* T para um conjunto $\mathcal{C}$ de cláusulas, denotada por $T_{\mathcal{C}}$, é uma árvore crescendo para baixo na qual cada aresta está associada a um conjunto finito de (negações de) átomos $A(\mathcal{C}) \subseteq H(\mathcal{C})$ de tal modo que:

(i) Para cada nó N existem apenas finitamente muitas arestas $L_1, ..., L_n$ de N. Para $\mathscr{C}_i$ a *conjunção* de todos os literais no conjunto associado a $L_i, i = 1, ..., n$, $\mathscr{C}_1 \vee \mathscr{C}_2 \vee ... \vee \mathscr{C}_n$ é uma fórmula proposicional válida.

(ii) Para cada nó N, seja que $I(N)$ é a união de todos os conjuntos associados às arestas do ramo de $T_{\mathcal{C}}$ descendo até e incluindo N. Então $I(N)$ não contém nenhum par complementar.

(2) Seja que $A(\mathcal{C}) = \{A_1, A_2, ..., A_k, ...\}$. Uma árvore semântica $T_{\mathcal{C}}$ é *completa* sse, para todo o nó terminal N, $I(N)$ contém A_i ou $\neg A_i$ para $i = 1, 2, ...$ N é um *nó de insucesso* se $I(N)$ falsifica uma qualquer instância básica de alguma cláusula

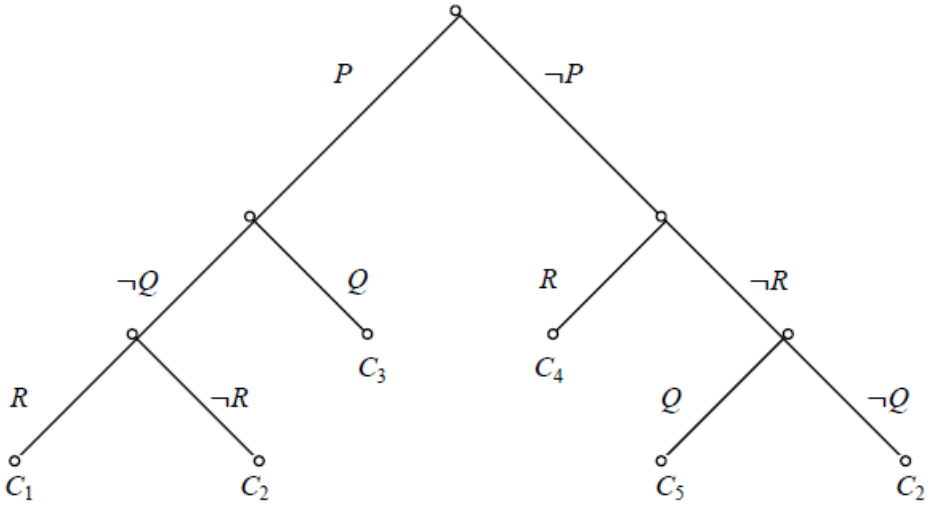


Figura 3.2.1: Árvore semântica fechada de $\mathcal{C} = \{C_1, C_2, C_3, C_4, C_5\}$.

C em $\mathcal{C}$, mas $I(N')$ não falsifica nenhuma instância de alguma cláusula de C em $\mathcal{C}$ para cada nó N' antecessor de N. Um ramo de uma árvore semântica $T_{\mathcal{C}}$ é *fechado* sse termina num nó de insucesso; caso contrário, é *aberto*. Uma árvore semântica $T_{\mathcal{C}}$ é *fechada* sse todos os seus ramos são fechados.

3.2.15. EXEMPLO. Sejam as seguintes cláusulas $C_1 = Q \vee \neg R$, $C_2 = Q \vee R$, $C_3 = \neg P \vee \neg Q$, $C_4 = P \vee \neg R$, $C_5 = P \vee \neg Q \vee R$ e seja $\mathcal{C}$ o conjunto das cláusulas $C_1 - C_5$; o conjunto de átomos de $\mathcal{C}$ é $A(\mathcal{C}) = \{P, Q, R\}$. As condições da def. 3.2.14 (1-2) são satisfeitas para uma árvore semântica fechada. Mostra-se a árvore semântica fechada de $\mathcal{C}$ na figura 3.2.1.

De volta à interpretação H, se $I(N)$, que é de facto uma interpretação parcial de $\mathcal{C}$ (i.e., $I(N)$ pode ser visto como uma atribuição de valores de verdade aos átomos básicos de $H(\mathcal{C})$), falsifica $\mathcal{C}$, então podemos parar de expandir nós a partir de N. Isto significa que se $\mathcal{C}$ é insatisfazível, a sua árvore semântica $T_{\mathcal{C}}$ não pode deixar de ser finita, eliminando-se assim o problema que se pusera acima.

3.2.16. EXEMPLO. Seja que $\mathcal{C} = \{\neg P(x) \vee Q(x), P(f(a)), \neg Q(z)\}$. Então,

$H_{\mathcal{C}} = \{a, f(a), f(f((a))), ...\}$
$H(\mathcal{C}) = \{P(a), Q(a), P(f(a)), Q(f(a)), ...\}$.

$\mathcal{C}$ é insatisfazível. A figura 3.2.2 mostra a árvore semântica fechada de $\mathcal{C}$. (Como é habitual, assinala-se um nó de insucesso com $\times$.)

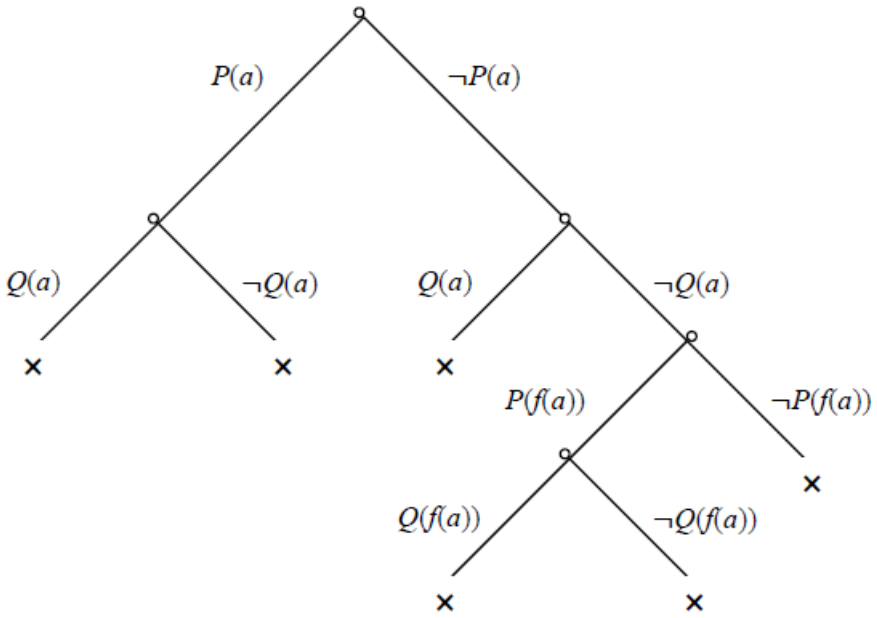


Figura 3.2.2: Árvore semântica fechada de $\mathcal{C} = \{\neg P(x) \vee Q(x), P(f(a)), \neg Q(z)\}$.

3.2.17. TEOREMA. (Herbrand, 1930 - versão I). Um conjunto $\mathcal{C}$ de cláusulas é insatisfazível sse correspondendo a cada árvore semântica de $\mathcal{C}$ existe uma árvore semântica finita fechada.

Demonstração. A demonstração segue-se imediatamente do que se desenvolveu acima:

($\Rightarrow$) Suponha-se que $\mathcal{C}$ é insatisfazível e que $T_{\mathcal{C}}$ é uma árvore semântica completa para $\mathcal{C}$. Para todo o ramo há o conjunto de etiquetas $I(N)$ que é uma interpretação, porque a árvore é completa. Logo, $I(N)$ falsifica uma instância básica C' de uma cláusula $C \in \mathcal{C}$. Visto que existem apenas finitamente muitos literais em C', deve haver um nó de insucesso N a uma distância finita da raiz. Uma vez que todos os ramos terminam num nó de insucesso existe uma árvore semântica correspondente a $T'_{\mathcal{C}}$ que é finita.

($\Leftarrow$) Suponha-se que para toda a árvore semântica completa $T_{\mathcal{C}}$ há uma árvore finita fechada $T'_{\mathcal{C}}$ correspondente. Então, todos os ramos terminam num nó de insucesso e logo todas as interpretações $I(N)$ falsificam $\mathcal{C}$. Logo, $\mathcal{C}$ é insatisfazível. □

A razão pela qual decidimos deixar para último lugar a versão I do teorema de Herbrand é que esta está mais diretamente associada à *resolução*, enquanto a versão

II tem mais a ver com *refutação*. Contudo, as duas versões tornam claro o facto que *a resolução é um procedimento de refutação* e uma demonstração da versão II pode ser bastante simplificada se pudermos já aplicar a definição de árvore semântica:

Demonstração. (Teorema de Herbrand, versão II)

($\Rightarrow$) Suponha-se que $\mathcal{C}$ é insatisfazível e que $T_{\mathcal{C}}$ é uma árvore semântica completa para $\mathcal{C}$. Então, pelo teorema de Herbrand, versão I, existe uma árvore semântica fechada finita $T'_{\mathcal{C}}$ de $\mathcal{C}$. Seja $\mathcal{C}'$ o conjunto de instâncias básicas que são falsificadas em todos os nós de insucesso de $T'_{\mathcal{C}}$: $\mathcal{C}'$ é finito e é falsificado por todas as interpretações. Segue-se que $\mathcal{C}'$ é insatisfazível.

($\Leftarrow$) A demonstração é por contraposição. □

Pode-se tornar a versão II do teorema de Herbrand num procedimento de demonstração bastando para tal gerar sucessivamente os conjuntos $\mathcal{C}'_0, \mathcal{C}'_1, ...$ em que $\mathcal{C}'_i$ é o conjunto de todas as instâncias básicas de cláusulas de $\mathcal{C}$ e testá-las em termos de insatisfazibilidade pelos meios disponíveis no cálculo proposicional: o teorema diz-nos que para N finito há um conjunto $\mathcal{C}'_N$ que é insatisfazível se $\mathcal{C}$ é insatisfazível. Gilmore (1960) foi o primeiro a implementar esta ideia com o método da multiplicação: à medida que cada $\mathcal{C}'_i$ é gerado, é multiplicado numa FND; uma qualquer conjunção na FND que contenha um par complementar é removida e se algum $\mathcal{C}'_i$ é vazio, então encontrou-se uma demonstração da sua insatisfazibilidade. Porém, como se disse acima, este método é bastante ineficaz, uma vez que para um conjunto de dez cláusulas básicas de dois literais, por exemplo, há 2^{10} conjunções.

3.3 O princípio de resolução

3.3.1 O princípio de resolução para a lógica proposicional

Davis e Putnam (1960) solucionaram esta ineficácia computacional por meio das quatro regras mencionadas acima. É aqui necessária uma exposição breve sobre uma delas, a *regra do literal único*, visto que o princípio de resolução é uma extensão desta regra. Esta regra asserta que (i) se houver uma cláusula básica unitária L em $\mathcal{C}$ podemos obter $\mathcal{C}'$ a partir de $\mathcal{C}$ pela remoção das cláusulas básicas de $\mathcal{C}$ que contêm L. (ii) Se $\mathcal{C}'$ é vazio, então $\mathcal{C}$ é satisfazível. (iii) Se $\mathcal{C}'$ não é vazio, obtemos um conjunto $\mathcal{C}''$ a partir de $\mathcal{C}'$ pela remoção de $\neg L$ de $\mathcal{C}'$. $\mathcal{C}''$ é insatisfazível sse $\mathcal{C}$ o é.

3.3.1. EXEMPLO. Seja $\mathcal{C} = \{\neg P \vee Q \vee \neg R, \neg P \vee \neg Q, P, R, U\}$. Aplicando a regra do literal único, condição (i) para P, obtém-se $\mathcal{C}' = \{\neg P \vee Q \vee \neg R, \neg P \vee \neg Q, R, U\}$; por (iii) – uma vez que $\mathcal{C}'$ não é vazio – obtém-se $\mathcal{C}'' = \{Q \vee \neg R, \neg Q, R, U\}$. Temos que $\mathcal{C} \equiv_{sat} \mathcal{C}''$. Pode-se agora repetir o processo para $\mathcal{C}''$: visto que R satisfaz a condição (i), obtém-se $\mathcal{C}''' = \{Q, \neg Q, U\}$. De acordo com a regra, $\mathcal{C} \equiv_{sat} \mathcal{C}'''$. Visto que $Q \wedge \neg Q = \square$, tem-se que $\mathcal{C}'''$ contém a cláusula vazia e, logo, $\mathcal{C}$ é insatisfazível (cf. prop. 1.1.23 e prop. 1.1.32).

Fazemos notar que estamos já a trabalhar com uma noção de insatisfazibilidade que tem imediatamente a ver com refutação, como se pode facilmente verificar pelo teorema seguinte.

3.3.2. TEOREMA. Uma fórmula F é insatisfazível sse é possível derivar uma contradição de F, i.e., $F \models G \wedge \neg G$.

Demonstração. Pelo teorema da dedução (teorema 1.2.8-9), temos que $F \models G \wedge \neg G$ sse $\models F \rightarrow (G \wedge \neg G)$. Por seu turno, temos que $\models F \rightarrow (G \wedge \neg G)$ sse, para toda a interpretação $\mathcal{I}$, (i) $val_{\mathcal{I}}(F) = \mathrm{F}$ ou (ii) $val_{\mathcal{I}}(F) = \mathrm{V}$ e $val_{\mathcal{I}}(G \wedge \neg G) = \mathrm{V}$. Mas pela proposição 1.1.23 (*) $(G \wedge \neg G) \in CONT(\mathrm{C}_2)$, pelo que $val_{\mathcal{I}}(G \wedge \neg G) = \mathrm{F}$ para toda a interpretação $\mathcal{I}$, pelo que $\models F \rightarrow (G \wedge \neg G)$ sse $val_{\mathcal{I}}(F) = \mathrm{F}$ para toda a interpretação $\mathcal{I}$. Logo, $\models F \rightarrow (G \wedge \neg G)$ sse F é insatisfazível, e temos então que $F \models (G \wedge \neg G)$ sse F é insatisfazível. □

A regra do literal único funda-se precisamente nesta noção de insatisfazibilidade de acordo com a qual um conjunto de cláusulas $\mathcal{C}$ é insatisfazível sse conseguirmos derivar de $\mathcal{C}$ a cláusula vazia (cf. prop. 1.1.23, (*) e (**)). Estendendo esta regra e aplicando-a a um qualquer par de cláusulas (não necessariamente unitárias)[4] obtemos o *princípio de resolução*:

Para quaisquer duas cláusulas C_1 e C_2 e dois literais complementares $L_1 \in C_1$ e $L_2 \in C_2$, elimina-se L_1 e L_2 de C_1 e C_2, respetivamente, e constrói-se a disjunção das restantes cláusulas, i.e., $C_1 \vee C_2$.

3.3.2. DEFINIÇÃO. A cláusula construída $C_1 \vee C_2$ é a *resolvente* de C_1 e C_2.

[4]Rejeitando-se assim a restrição imposta pela regra do literal único.

3.3.3. Teorema. Uma resolvente $C = C_1' \vee C_2'$ de duas cláusulas $C_1 = L \vee C_1'$ e $C_2 = L \vee C_2'$ é uma consequência lógica de $C_1 \wedge C_2$, i.e.,

$$\frac{C_1' \vee L \quad C_2' \vee \neg L}{C_1' \vee C_2'}.$$

Demonstração. Sejam $C_1 = L \vee C_1'$, $C_2 = \neg L \vee C_2'$ e $C = C_1' \vee C_2'$, C_1' e C_2' são disjunções de literais. Supondo-se que C_1 e C_2 são ambas verdadeiras numa interpretação $\mathcal{I}$, a sua resolvente C deve de igual modo ser verdadeira em $\mathcal{I}$. Obviamente, L ou $\neg L$ é falso em $\mathcal{I}$. Suponha-se que L é falso em $\mathcal{I}$; então C_1 não deve ser uma cláusula unitária, caso contrário seria falsa em $\mathcal{I}$. Logo, C_1' deve ser verdadeira em $\mathcal{I}$ e a resolvente $C_1' \vee C_2'$ é verdadeira em $\mathcal{I}$. Suponha-se que $\neg L$ é falso em $\mathcal{I}$ e proceda-se do mesmo modo. Logo, $C_1' \vee C_2'$ é verdadeira em $\mathcal{I}$. □

3.3.4. Definição. Uma *dedução por resolução* de C a partir de um conjunto de cláusulas $\mathcal{C}$, denotada por $\mathcal{C} \vdash_{res} C$, é uma sequência finita $C_1, C_2, ..., C_k$ de cláusulas tal que cada C_i é ou uma cláusula em $\mathcal{C}$ ou uma resolvente de cláusulas precedendo C_i e $C_k = C$. Dizemos que a dedução da cláusula vazia $\square$ a partir de $\mathcal{C}$ é uma *refutação* ou uma *demonstração* de $\mathcal{C}$.

A dedução da cláusula vazia a partir de um conjunto de cláusulas $\mathcal{C}$ pode ser representada por meio de uma *árvore de dedução* (ou *de refutação*).

3.3.5. Exemplo. Seja $\mathcal{C} = \{P \vee Q, \neg P \vee Q, P \vee \neg Q, \neg P \vee \neg Q\}$. A árvore de dedução de $\mathcal{C}$ mostra que $\mathcal{C}$ é insatisfazível (fig. 3.3.1).

3.3.2 O princípio de resolução para LPO

Abordamos agora o princípio de resolução para LPO. Com este fim em vista, precisamos de algumas noções básicas suplementares. (Damos aqui os aspetos essenciais; para uma formalização completa, ver, por exemplo, Chang & Lee (1973).)

3.3.6. Definição. Dado um conjunto $V = \{x_1, ..., x_n\}$ de variáveis e um conjunto $T = \{t_1, ..., t_n\}$ de termos, uma *substituição* é um mapeamento $\sigma : V \longrightarrow T$ tal que $\sigma(x_i) = t_i$ na generalidade e onde $t_i \neq x_i$. Representamos uma substituição σ como um conjunto finito de expressões com a forma $x_i \mapsto t_i$ (lê-se: x_i é substituída por

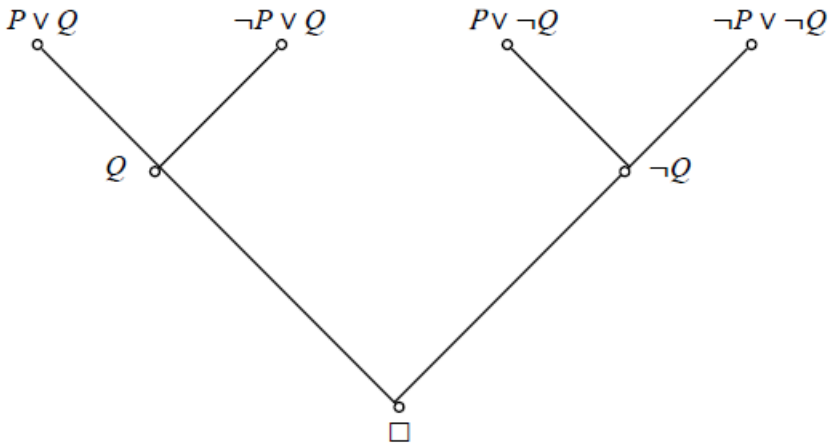


Figura 3.3.1: Árvore de dedução de $\mathcal{C} = \{P \vee Q, \neg P \vee Q, P \vee \neg Q, \neg P \vee \neg Q\}$.

t_i) em que dois termos diferentes nunca substituem a mesma variável, i.e.,

$$\sigma = \{x_1 \mapsto t_1, ..., x_n \mapsto t_n\}.$$

Definimos o domínio de σ como $\text{dom}(\sigma) = \{x \mid \sigma(x) \neq x\}$ e o seu contradomínio como $\text{ctdom}(\sigma) = \{\sigma(x) \mid x \in \text{dom}(\sigma)\}$. Dizemos que σ é uma *substituição básica* (*ground*) quando $V(\text{ctdom}(\sigma)) = \emptyset$, ou seja, quando $t_1, ...t_n$ são termos básicos. No caso de $\sigma = \emptyset$, falamos da *substituição vazia* e denotamo-la por ϵ. Uma substituição injetiva σ tal que $\text{ctdom}(\sigma) \subseteq V$ chama-se uma *renomeação (de variável).*

3.3.7. Definição. Para uma substituição $\theta = \{x_1 \mapsto t_1, ..., x_n \mapsto t_n\}$ e uma expressão E, $E\theta$ é uma expressão obtida a partir de E pela substituição simultânea de cada ocorrência da variável x_i, $1 \leq i \leq n$, em E pelo termo t_i. Dizemos que $E\theta$ é uma *instância* de E. Se $V(E\theta) = \emptyset$, então $E\theta$ chama-se uma *instância básica.*

3.3.8. Exemplo. Seja $\theta = \{x \mapsto a, y \mapsto f(b), z \mapsto c\}$ e $E = P(x, y, z)$. Então, $E\theta = P(a, f(b), c)$ e $E\theta$ é uma instância básica de E.

3.3.9. Definição. Dadas duas substituições $\theta = \{x_1 \mapsto t_1, ..., x_n \mapsto t_n\}$ e $\lambda = \{y_1 \mapsto u_1, ..., y_n \mapsto u_n\}$, a sua composição, denotada por $\theta \circ \lambda$, obtém-se a partir do conjunto $\{x_1 \mapsto t_1\lambda, ..., x_n \mapsto t_n\lambda, y_1 \mapsto u_1, ..., y_n \mapsto u_n\}$ pela remoção de todos os elementos $x_j \mapsto t_j\lambda$ para os quais $t_j\lambda = x_j$ e de todos os elmentos $y_i \mapsto u_i$ tais que y_i está entre $\{x_1, ..., x_n\}$.

3.3.10. Exemplo. Dados os conjuntos de substituição $\theta = \{x \mapsto f(y), y \mapsto z\}$

e $\lambda = \{x \mapsto a, y \mapsto b, z \mapsto y\}$, então $\theta \circ \lambda = \{x \mapsto f(b), z \mapsto y\}$.

3.3.11. DEFINIÇÃO (unificação). Dizemos que um conjunto de expressões $E = \{E_1, ..., E_n\}$ é *unificável* por uma substituição σ (o *unificador* para E) se $E_i\sigma = E_j\sigma$ para todas $E_i, E_j \in E$. Um unificador σ para o conjunto E é um *unificador mais geral (umg)* sse para cada unificador θ para o conjunto existe uma substituição λ tal que $\theta = \sigma \circ \lambda$. (Equivalentemente, seja que se diz que λ é mais geral do que θ, denotado por $\lambda \leq_s \theta$, se existe uma substituição σ tal que $\sigma \circ \lambda = \theta$. Então, σ é o umg do conjunto E se para qualquer outro unificador λ se verifica que $\sigma \leq_s \lambda$.)

3.3.12. EXEMPLO. $\theta = \{x \mapsto f(b), y \mapsto b, z \mapsto u\}$ é um unificador das expressões $E_1 = f(x, b, g(z))$ e $E_2 = f(f(y), y, g(u))$, porque $E_1\theta = E_2\theta = f(f(b), b, g(u))$.

3.3.13. EXEMPLO. Dadas as substituições $\theta = \{x \mapsto z, y \mapsto z, u \mapsto f(z, z)\}$, $\lambda = \{y \mapsto z\}$ e $\sigma = \{x \mapsto y, u \mapsto f(y, z)\}$, então σ é o umg porque $\theta = \sigma \circ \lambda$.

3.3.14. DEFINIÇÃO. Sejam C_1 e C_2 duas cláusulas. Então, $C_1 \leq_{ss} C_2$ se existir uma substituição σ tal que $C_1\sigma \subseteq C_2$ e dizemos que C_1 *subsume* C_2.

3.3.15. DEFINIÇÃO. Dado um conjunto não vazio de expressões E, o *conjunto de desacordo* $D(E)$ de E é o conjunto das subexpressões de E obtidas localizando o símbolo mais à esquerda no qual nem todas as expressões em E têm exatamente o mesmo símbolo e extraindo de cada expressão em E a subexpressão que começa com o símbolo que ocupa essa posição.

3.3.16. EXEMPLO. Para $E = \{P(x, f(y, z)), P(x, a), P(x, g(h(k(x))))\}$ temos que $D(E) = \{f(y, z), a, g(h(k(x)))\}$.

O *problema da unificação* é o de encontrar um umg de dois dados termos. Este é um procedimento puramente mecânico para o qual há mais do que um algoritmo, de entre os quais o *algoritmo de Robinson* parece ser o mais eficiente (cf. Hoder & Voronkov, 2009). Dado um conjunto *Lit* de literais como *input*,

Faça-se $\sigma := \epsilon$
Enquanto $|\sigma(Lit)| > 1$ {
selecionar um par de desacordo d em $\sigma(Lit)$;
se *d não contém nenhuma variável* **então** {

stop e **retorne-se** "não unificável";
} **caso contrário** {
faça-se $d = (x, t)$ com x uma variável;
se x ocorre em t, **então** "*occurs check*" {
stop e **retorne-se** "não unificável";
} **caso contrário** {
faça-se $\sigma := \sigma \circ (x \mapsto t)$;
}
}
} **retorne-se** σ;

A aplicação de *regras de unificação* é um procedimento alternativo (cf. por exemplo, Baader & Snyder, 2001). Sejam σ e θ designações para substituições e denote-se que a aplicação de uma regra *falha* por $\perp$. Para P e Q pares de expressões $\{\langle E_1, F_1\rangle, ..., \langle E_n, F_n\rangle\}$, as regras de unificação de P e Q têm a forma geral

$$P; \sigma \Rightarrow Q; \theta \quad \text{ou}$$

$$P; \sigma \Rightarrow \perp.$$

O procedimento de aplicação (sucessiva) das regras de unificação termina ou em sucesso (denotado por $\emptyset$) ou em falha. Note-se que a ordem de aplicação das regras não é determinística.

As regras de inferência específicas são as seguintes:

1. *Trivial*:

$$\{\langle s, s\rangle\} \cup P' : \sigma \Rightarrow P'; \sigma$$

Exemplo: Para o par de expressões $\langle P(a), P(a)\rangle$, temos

$\{\langle P(a), P(a)\rangle\}; \epsilon \Rightarrow_{Tr}$
$\emptyset; \epsilon$

2. *Decomposição*:

$$\{\langle f(s_1, ..., s_n), f(t_1, ..., t_n)\rangle\} \cup P'; \sigma \Rightarrow \{\langle s_1, t_1\rangle, ..., \langle s_n, t_n\rangle\} \cup P'; \sigma$$

se $f(s_1, ..., s_n) \neq f(t_1, ..., t_n)$.

Exemplo: Para o par de expressões $P(f(a), g(x))$ e $P(y, z)$ temos

$\{\langle P(f(a), g(x)), P(y, z)\rangle\}; \epsilon \Rightarrow_{Dec}$
$\{\langle f(a), y\rangle, \langle g(x), z\rangle\}; \epsilon$

3. *Orientação*:

$$\{\langle t, x\rangle\} \cup P'; \sigma \Rightarrow \{\langle x, t\rangle\} \cup P'; \sigma$$

na condição de t não ser uma variável.

Exemplo: Aplicando esta regra ao resultado obtido no exemplo acima, temos

$\{\langle f(a), y\rangle, \langle g(x), z\rangle\}; \epsilon \Rightarrow_{Or}$
$\left\{\left\langle \underline{y, f(a)} \right\rangle, \langle g(x), z\rangle\right\}; \epsilon$

4. *Eliminação de variáveis*:

$$\{\langle x, t\rangle\} \cup P'; \sigma \Rightarrow P'\theta; \sigma\theta$$

na condição de x não ocorrer em t e $\theta = \{x \rightarrow t\}$.

Exemplo: Dado o par de expressões $P(x, f(a))$ e $P(g(y), x)$, temos

$\{\langle P(x, f(a)), P(g(y), x)\rangle\}; \epsilon \Rightarrow_{EV}$
$\{\langle g(y), f(a)\rangle\}; \{x \rightarrow f(a)\}$

5. *Choque de símbolos*:

$$\{\langle f(s_1, ..., s_n), g(t_1, ..., t_m)\rangle\} \cup P'; \sigma \Rightarrow \perp$$

se $f \neq g$.

Exemplo: Tomando o resultado obtido no exemplo anterior, temos

$\{\langle g(y), f(a)\rangle\}; \epsilon \Rightarrow_{CHS}$
$\perp$

6. *Occurs check*:

$$\{\langle x, t\rangle\} \cup P'; \sigma \Rightarrow \bot$$

se x ocorre em t e $x \neq t$.

Exemplo: Dado o par de expressões $P(x)$ e $P(f(x))$, temos

$\{\langle x, f(x)\rangle\}; \epsilon \Rightarrow_{OC}$
$\bot$

3.3.17. DEFINIÇÃO (resolução binária). Sejam C_1 e C_2 duas cláusulas (chamadas *cláusulas pais*) sem quaisquer variáveis em comum. Sejam L_1 e L_2 dois literais em C_1 e C_2, respetivamente. Então, a cláusula

$$(C_1 - L_1)\sigma \cup (C_2 - L_2)\sigma$$

em que σ é um umg, chama-se uma *resolvente binária* de C_1 e C_2, e os literais L_1 e L_2 são os *literais resolvidos*. A regra de inferência é

$$\frac{C_1 \vee L_1 \quad C_2 \vee \neg L_2}{(C_1 \vee C_2)\sigma}$$

em que σ é um umg de L_1 e L_2.

3.3.18. DEFINIÇÃO (fatorização). Um *fator* de uma cláusula C é uma cláusula $C\sigma$ em que σ é um umg de uma qualquer $C' \subseteq C$. Se $C\sigma$ é uma cláusula unitária, então chama-se um *fator unitário* de C. Note-se que toda a cláusula é um fator de si mesma, i.e.,

$$\frac{C}{C\sigma}.$$

Dadas n cláusulas $C_1, ..., C_n$, a seguinte regra de inferência chama-se *fatorização (positiva)*:

$$\frac{C_1 \vee C_2 \vee C_3}{(C_1 \vee C_2)\sigma}$$

em que σ é um umg de C_2 e C_3.

3.3.19. DEFINIÇÃO. Uma resolvente das cláusulas (pais) C_1 e C_2 é uma das seguintes resolventes binárias:

(i) uma resolvente binária de C_1 e C_2,
(ii) uma resolvente binária de C_1 e um fator de C_2,
(iii) uma resolvente de um fator de C_1 e C_2,
(iv) uma resolvente de um fator de C_1 e um fator de C_2.

3.3.20. Teorema (resolução binária). Uma resolvente $C = C_1' \vee C_2'$ de duas cláusulas (pais) $C_1 = C_1' \vee L_1$ e $C_2 = C_2' \vee L_2$ de LPO é uma consequência lógica de C_1 e C_2 se existe uma substituição σ tal que σ unifica o par de literais complementares L_1 e L_2, i.e.,

$$\frac{C_1' \vee L_1 \quad C_2' \vee L_2}{\left(C_1' \vee C_2'\right)\sigma}.$$

3.3.21. Exemplo. Sejam $C_1 = P(x) \vee Q(x)$ e $C_2 = \neg P(z) \vee R(x) \vee \neg P(a)$. De modo a implementar a resolução binária nestas duas cláusulas, temos de (i) renomear a variável x em C_2 (cf. def. 3.3.6), uma vez que esta aparece nas duas cláusulas, bastando para tal renomear x como y; (ii) fatorizar $\neg P(z)$ e $\neg P(a)$ em C_2, nomeadamente por meio do conjunto de substituição $\sigma = \{z \mapsto a\}$ (cf. def. 3.3.18). Temos então $C_1 = P(x) \vee Q(x)$ e $C_2\sigma = \neg P(a) \vee R(y)$. Dado o umg $\theta = \{x \mapsto a\}$, temos a resolvente binária $Q(x) \vee R(y)$ correspondente a $\left(C_1' \vee C_2'\sigma\right)\theta$ para os literais resolvidos $L_1 = P(x)$ e $L_2 = \neg P(a)$. A figura 3.3.2 mostra como a árvore de dedução falha: $\mathcal{C} = \{C_1, C_2\}$ é satisfazível, pelo que não se pode deduzir de $\mathcal{C}$ a cláusula vazia.

3.3.3 Completude do princípio de resolução

Duas regras de inferência descrevem o cálculo de resolução de modo essencial: a resolução binária (def. 3.3.17) e a fatorização (positiva) (def. 3.3.18). De modo a demonstrar a completude do princípio de resolução necessitamos ainda do teorema conhecido habitualmente por *lifting lemma*:

3.3.22. Teorema (*Lifting lemma*). Sejam C_1', C_2' instâncias de C_1 e C_2, respetivamente. Se C'é uma resolvente de C_1' e de C_2', então existe uma resolvente C de C_1 e C_2 tal que C' é uma instância de C.

Demonstração. (Esboço) Seja

$$C' = \left(C_1'\gamma - L_1'\gamma\right) \cup \left(C_2'\gamma - L_2'\gamma\right)$$

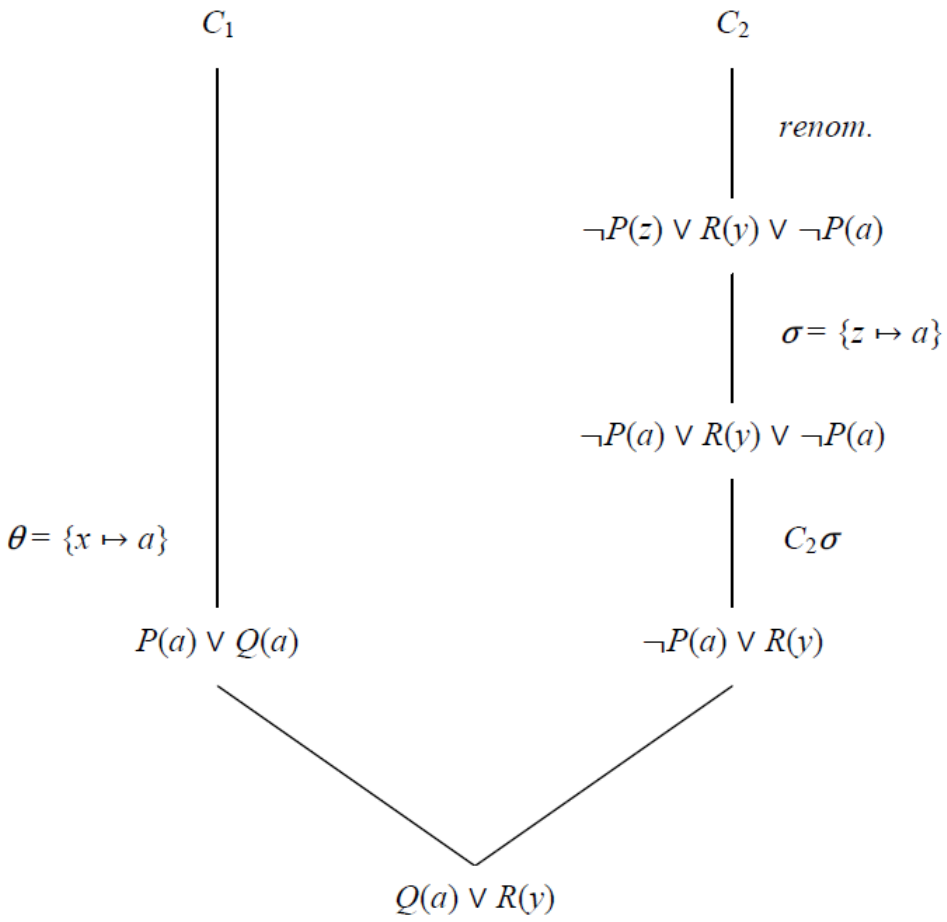


Figura 3.3.2: Exemplo de uma árvore de dedução que falha.

em que γ é um umg de $L_1^{'}$ e $\neg L_2^{'}$. Tem de se mostrar que $C^{'}$ é uma instância de C,

$$C = ((C_1\lambda)\,\sigma - L_1\sigma) \cup ((C_2\lambda)\,\sigma - L_2\sigma)$$

em que σ é um umg de L_1 e $\neg L_2$ e λ_i é um umg para $\{L_i^1, ..., L_i^{r_i}\}$ com $L_i = L_i^1\lambda_i$, $i = 1, 2$, se $r_i > 1$; se $r_i = 1$, então faz-se $\lambda_i = \epsilon$ e $L_i = L_i^1\lambda_i$. Para tal, basta mostrar que existe uma substituição θ tal que $C_1^{'} = C_1\theta$, $C_2^{'} = C_2\theta$ e $(\lambda \circ \sigma) \leq_s (\theta \circ \gamma)$. □

3.3.23. Teorema (completude do princípio de resolução). Um conjunto $\mathcal{C}$ de cláusulas é insatisfazível sse existe uma dedução de $\mathcal{C}$ da cláusula vazia □.

Demonstração. ($\Rightarrow$) Damos um esboço da demonstração. Dado que uma contradição (i.e., a cláusula vazia) pode ser deduzida a partir de um qualquer conjunto de cláusulas insatisfazível, a procura por uma é por saturação do conjunto de cláusulas, ou seja, pela aplicação sistemática e exaustiva das regras de inferência até derivar a cláusula vazia. Em termos de uma árvore semântica, isto implica que se gera a árvore para um conjunto $\mathcal{C}$ só com o nó-raiz após, pelo teorema 3.3.22, um processo de obtenção de árvores cada vez mais pequenas $T_{\mathcal{C}}^{'}, T_{\mathcal{C}}^{''}, ...$ para $\mathcal{C} \cup \{C\}$, em que C é uma resolvente das cláusulas C_1 e C_2. O nó-raiz só é gerado quando se deriva □. Logo, existe uma dedução de □ a partir de $\mathcal{C}$.

($\Leftarrow$) Suponha-se que existe uma dedução de □ a partir de $\mathcal{C}$. Sejam $R_1, ..., R_k$ as resultantes na dedução. Suponha-se que $\mathcal{C}$ é satisfazível. Então, existe um modelo M de $\mathcal{C}$. Pelo teorema 3.3.3, M satisfaz $R_1, ..., R_k$. Mas isto é impossível, uma vez que uma destas resolventes é □. Logo, $\mathcal{C}$ deve ser insatisfazível. □

3.4 Refinamentos da resolução

Os aspetos fundamentais da resolução foram dados acima. De modo a evitar a formação de cláusulas redundantes aplicam-se vários refinamentos da resolução.

3.4.1. Definição. Sejam Res_{rf} um mapeamento do conjunto $\mathbf{C}$ de todos os conjuntos finitos de cláusulas para o conjunto Res de todas as deduções por resolução (i.e., deduções R) e ϱ uma Res básica (i.e., *ground*). Res_{rf} é um *refinamento da resolução* sse

1. Para todo o conjunto de cláusulas $\mathcal{C} \in \mathbf{C}$ se tem que $Res_{rf}(\mathcal{C}) \subseteq Res(\mathcal{C})$.
2. $\{\varrho \mid \varrho \in Res_{rf}(\mathcal{C})\}$ é decidível para todo $\mathcal{C} \in \mathbf{C}$.
3. Para cada $\mathcal{C} \in \mathbf{C}$ existe um algoritmo ξ que constrói $Res_{rf}(\mathcal{C})$.

4. Se $\mathcal{C} \subseteq \mathcal{D}$, então tem-se que $Res_{rf}(\mathcal{C}) \subseteq Res_{rf}(\mathcal{D})$.

3.4.2. DEFINIÇÃO (completude de um refinamento de resolução). Um refinamento de resolução Res_{rf} é *completo* se, dado um qualquer conjunto de cláusulas insatisfazível $\mathcal{C}$, $Res_{rf}(\mathcal{C})$ contém uma refutação R de $\mathcal{C}$.

Há hoje muitos refinamentos da resolução (cf. Chang & Lee, 1973; Leitsch, 1997); abordamos aqui apenas aqueles de interesse para a resolução aplicada às lógicas multivalentes, a temática do próximo capítulo.

3.4.1 Ordenamento de átomos

3.4.3. DEFINIÇÃO (*A-ordering*). Sejam Ω o conjunto de todas as fórmulas atómicas e σ uma substituição qualquer. Um *ordenamento atómico* $<_A$ é uma relação binária em Ω tal que $<_A$ é (1) irreflexivo e (2) transitivo, e (3) para todo $A, B \in \Omega$ temos que $A < B$ implica que $A\sigma <_A B\sigma$.

Dada a condição (1), é óbvio que A e B não são unificáveis, i.e., $A\sigma \neq B\sigma$. O aspeto mais importante no que respeita a $<_A$ é a especificação do ordenamento; esta recai comummente na complexidade dos termos e das variáveis de átomos.

3.4.4. DEFINIÇÃO. (i) Denotamos por $\vartheta(\centerdot)$ e definimos *profundidade de termo*

(1) de um termo como

$$\vartheta(t) = 0$$

para $t \in V \cup Cons$; para $f \in Fun$, $t_1, ..., t_n$ são termos, temos

$$\vartheta(f(t_1, ..., t_n)) = 1 + max\{\vartheta(t_i) \mid i = 1, ..., n\};$$

(2) de um literal L

$$\vartheta(L) = max\{\vartheta(t) \mid t \in \arg(L)\},$$

(3) de uma cláusula C

$$\vartheta(C) = max\{\vartheta(L) \mid L \in C\},$$

(4) de um conjunto de cláusulas $\mathcal{C}$

$$\vartheta(\mathcal{C}) = max\{\vartheta(C) \mid C \in \mathcal{C}\}$$

(ii) Denotamos por $\vartheta_{max}(x, E)$ e definimos *profundidade maximal de uma variável x numa expressão E* como

(1) para E um termo t

$$\vartheta_{max}(x,t) = \begin{cases} 0 & \text{se } x \notin V(t) \text{ ou } x = t \\ 1 + max\{\vartheta_{max}(x, t_i) \mid i = 1, ..., n\} & \text{se } \blacklozenge \end{cases},$$

$$\blacklozenge = x \in V(t) \text{ e } t = f(t_1, ..., t_n), f \in Fun,$$

(2) para E um átomo $P(t_1, ..., t_n)$

$$\vartheta_{max}(x, P(t_1, ..., t_n)) = max\{(x, t_i) \mid i = 1, ..., n\},$$

(3) para E um literal L

$$\vartheta_{max}(x, L) = \vartheta_{max}(x, \arg(L)),$$

(4) para E uma cláusula $C = L_1 \vee ... \vee L_n$

$$\vartheta_{max}(x, L_1 \vee ... \vee L_n) = max\{\vartheta_{max}(x, L_i) \mid i = 1, ..., n\}.$$

3.4.5. EXEMPLO. Sejam $A = P(x, f(f(y)))$, $B = Q(f(x))$ e $C = \{A, \neg B\}$. Então, $\vartheta(A) = 2$, $\vartheta(B) = 1$, $\vartheta(C) = 2$, $\vartheta_{max}(x, C) = 1$ e $\vartheta_{max}(y, C) = 2$.

Dada esta definição, podemos agora especificar um ordenamento $<_a$ tal que, para quaisquer átomos A e B se tem que $A <_\alpha B$ sse

(i) $\vartheta(A) < \vartheta(B)$ e
(ii) para todo $x \in V(A) : \vartheta_{max}(x, A) < \vartheta_{max}(x, B)$ (implicando que $V(A) \subseteq V(B)$).

(i) e (ii) garantem a reflexividade e a transitividade de $<_\alpha$. Note-se que se (i) e (ii) são satisfeitas, então para todas as substituições σ e para todo $y \in V(A\sigma)$

temos que

(i*) $\vartheta(A\sigma) < \vartheta(B\sigma)$ e
(ii*) $\vartheta_{max}(y, A\sigma) < \vartheta_{max}(y, B\sigma)$.

3.4.6. EXEMPLO. Para $<_\alpha$ temos

$$P(x,x) <_\alpha Q(f(x), y)$$

uma vez que se tem que $\vartheta(P(x,x)) = 0$, $\vartheta(Q(f(x), y)) = 1$, $\vartheta_{max}(P(x,x)) = 0$ e $\vartheta_{max}(Q(f(x), y)) = 1$, e para qualquer substituição $\sigma = \{x \mapsto t\}$ para um qualquer termo t, $\vartheta(P(x,x)\sigma) < \vartheta(Q(f(x), y)\sigma)$ e $\vartheta_{max}(P(x,x)\sigma) < \vartheta_{max}(Q(f(x), y)\sigma)$.

3.4.7. EXEMPLO. Para $<_\alpha$ temos

$$P(x, f(a)) \not<_\alpha Q(x, f(x))$$

$$P(x, a) \not<_\alpha P(f(a), x)$$

No primeiro caso, a condição (i) é violada; no segundo caso, (ii) é violada.

3.4.8. DEFINIÇÃO. Chame-se *condensada* uma cláusula C se não existe nenhum fator de C que é uma subclasse própria de C e chame-se C' a *condensação de* C se C' é um fator de C tal que $C' \subseteq C$. As condensações são únicas a menos de renomeação. A *regra de condensação* estipula que num procedimento de demontração uma cláusula é imediatamente substituída pela sua condensação, se ela existir.

3.4.9. EXEMPLO. $\{P(x,y), P(y,x)\}$ é condensada. $\{P(x,y), P(x,a)\}$ não é condensada; a sua condensação é $\{P(x,a)\}$.

3.4.10. DEFINIÇÃO. Sejam $\mathcal{C}$ um conjunto de cláusulas condensadas e $<_A$ um ordenamento de átomos. Seja C uma resolvente de duas cláusulas $C_1, C_2 \in \mathcal{C}$. Então (a condensação de) C é uma *resolvente* $<_A$ de C_1 e C_2 se não existir nenhum literal L em C tal que $B <_A L$ para B o átomo resolvido na resolução de C_1 e C_2. Denotamos que C é uma resolvente $<_A$ de um conjunto de cláusulas $\mathcal{C}$ por $C \in \rho <_A (\mathcal{C})$.

3.4.11. EXEMPLO. Sejam C_1 e C_2 as cláusulas $C_1 = \{Q(f(x_1), x_1), \neg R(f(x_1))\}$ e $C_2 = \{R(f(x_1)), \neg Q(x_1, x_2)\}$. Obviamente, C_1 e C_2 são condensadas. Seja agora

$\mathcal{C} = \{C_1, C_2\}$. Queremos obter uma resolvente $C \in \rho <_\alpha (\mathcal{C})$.

O primeiro passo é uma renomeação das variáveis. Por exemplo, $C_1' = Q(f(x), x) \vee \neg R(f(x))$ e $C_2' = R(f(y)) \vee \neg Q(y, z)$. Temos duas possibilidades para obter uma resolvente $C \in \rho <_\alpha (\mathcal{C})$: Dado $\sigma = \{y \mapsto z\}$, $\sigma = umg\left(C_1', C_2'\right)$, obtemos $A = Q(f(x), x) \vee \neg Q((x, z))$ e com $\theta = \{y \mapsto f(x), z \mapsto x\}$, $\theta = umg\left(C_1', C_2'\right)$, obtemos $B = \neg R(f(x)) \vee R(f(f(x)))$. Porém, só para A se verifica que $A \in \rho <_\alpha (\mathcal{C})$, visto que $R(f(x)) \not<_\alpha L$ para $L = Q(f(x), x)$ ou $L = \neg Q(x, z)$; pelo contrário, $B \notin \rho <_\alpha (\mathcal{C})$ porque $Q(f(x), x) <_\alpha R(f(f(x)))$.

Note-se que este exemplo mostra que é necessário utilizar critérios *a posteriori* e que de entre estes temos de escolher o mais forte. Com efeito, um critério *a priori* de $<_\alpha$ não teria impedido a resolução de B, até porque *a priori* não existe qualquer relação de ordenamento $<_\alpha$ entre os átomos de C_1 e C_2.

3.4.12. DEFINIÇÃO. Seja $\mathcal{C}$ um conjunto de cláusulas condensadas e $<_A$ um ordenamento de átomos. Uma *dedução* $Res <_A$ de uma cláusula condensada C de $\mathcal{C}$ é uma sequência $C_1, ..., C_n$ tal que

(1) $C_n = C$ e
(2) Para todo $i = 1, ..., n$, $C_i \in \mathcal{C}$ ou $C_i \in \rho <_A (\{C_j, C_k\})$ para quaisquer $j, k < i$.

3.4.13. TEOREMA (completude da dedução $Res <_A$). Seja $\mathcal{C}$ um conjunto finito de cláusulas condensadas e $<_A$ um ordenamento de átomos. Se $\mathcal{C}$ é insatisfazível, então existe uma refutação $Res <_A$ de $\mathcal{C}$.

Demonstração. ($\Rightarrow$) A demonstração é por construção de uma árvore semântica fechada: dado o conjunto de átomos $A(\mathcal{C}) = \{A_1, ..., A_n\}$ e um ordenamento tal que $A_i < A_j$ para $i < j \leq n$, é possível construir uma árvore com as etiquetas A_1 e $\neg A_1$ nas duas arestas que partem imediatamente do nó-raiz e com as etiquetas A_n e $\neg A_n$ nas arestas dos nós de insucesso, sendo que em cada ramo existe um nó de insucesso. Lembramos que pelo teorema 3.2.17 cada nó de insucesso falsifica uma cláusula de $\mathcal{C}$ e que uma árvore semântica $T_\mathcal{C}$ é fechada se todos os ramos terminam num nó de insucesso. Se tomarmos uma árvore semântica fechada de $\mathcal{C}$, revertendo o processo da sua construção até ao seu colapso na cláusula vazia (a raiz), ou seja, $\square \in Res <_A$, mostramos que não pode deixar de existir uma dedução $Res <_A$ de $\mathcal{C}$. (Cf. teorema 3.3.23 e respetiva demonstração.)

($\Leftarrow$) A demonstração segue como para o teorema 3.3.23, i.e., nomeadamente pela definição 3.4.10 e pelo facto que $\square \in Res <_A (\mathcal{C})$ para uma qualquer $C_i = \square$. $\square$

3.4.2 Hiper-resolução e resolução semântica

A *hiper-resolução* é uma espécie de *macro-resolução*, ou seja, a contração de uma sequência de passos de resolução numa única inferência.

3.4.14. DEFINIÇÃO. Sejam C uma cláusula não positiva e $D_1, ..., D_n$ cláusulas positivas, $C, D_1, ..., D_n \in \mathcal{C}$. Então, $\Xi = (C; D_1, ..., D_n)$ chama-se uma *sequência de choque* em que C é o *núcleo* e $D_1, ..., D_n$ são os *satélites*. Sejam $C_0 = C$ e $C_{i+1} \in Res(\{C_i, D_{i+1}\})$ para $i = 1, ..., n-1$. Se C_n é definida e positiva, então diz-se que C_n é uma *hiper-resolvente* de Ξ. Denotamos por $R_H(\mathcal{C})$ o conjunto de todas as hiper-resolventes de um conjunto de cláusulas $\mathcal{C}$ e chamamos *operador de hiper-resolução* ao operador correspondente R_H^*.

3.4.15. EXEMPLO. Seja $\mathcal{C}$ o conjunto das cláusulas $C_1 = P(a,b)$, $C_2 = P(b,a)$, $C_3 = \neg P(x,y) \vee \neg P(y,z) \vee P(x,z)$ e $C_4 = \neg P(a,a)$. Na refutação por resolução que se segue, uma das cláusulas resolventes é sempre positiva; com efeito, trata-se de resolver $\Xi = (C_3; C_1, C_2)$, em que os satélites (ou eletrões) C_1 e C_2 são cláusulas positivas.

C_1	$P(a,b)$	
C_2	$P(b,a)$	
C_3	$\neg P(x,y) \vee \neg P(y,z) \vee P(x,z)$	
C_4	$\neg P(a,a)$	
C_5	$\neg P(x,b) \vee P(x,a)$	Resolvente de $(C_2, C_3)\,\sigma$, $\sigma = \{y \mapsto b, z \mapsto a\}$
C_6	$P(a,a)$	Resolvente de $(C_1, C_5)\,\theta$, $\theta = \{x \mapsto a\}$
C_7	$\square$	Resolvente de C_4 e C_6

Dizemos que $C_6 = P(a,a)$ é uma *hiper-resolvente* de $\Xi = (C_3; C_1, C_2)$ na medida em que se pode dizer que $C_5 = \neg P(x,b) \vee P(x,a)$ é um resultado intermédio com relação a C_6. Note-se que em C_5 o literal negativo pertence ao núcleo e o positivo é de facto o satélite $(C_1)\,\lambda$ (ou$(C_2)\,\lambda$) para $\lambda = \{b \mapsto a\}$. As *macro-resolventes* de $\mathcal{C}$ são pois C_6 e C_7. Temos então que $R_H^*(\mathcal{C}) = \mathcal{C} \cup \{P(a,a), \square\}$, sendo que todas as cláusulas produzidas contêm no máximo um átomo.

Este é um exemplo de *hiper-resolução positiva* porque todos os eletrões e as hiper-resolventes são positivas. No caso de *hiper-resolução negativa* os eletrões e as hiper-

resolventes são negativas. Num caso como no outro, trata-se de impor uma interpretação na resolução de um conjunto de cláusulas, pelo que a hiper-resolução é um tipo de *resolução semântica.*

3.4.16. DEFINIÇÃO. Dado um conjunto de cláusulas $\mathcal{C}$ e o conjunto $Pred(\mathcal{C})$ dos predicados de $\mathcal{C}$, seja $\mathcal{I} = (\mathscr{D}, \Theta, \delta)$ uma interpretação tal que para todo predicado de m lugares $P \in Pred(\mathcal{C})$ temos que $\Theta(P)(d_1, ..., d_m) = \mathrm{F}$ para todo $d_1, ..., d_m \in \mathscr{D}$. Então todas as cláusulas positivas são falsas em $\mathcal{I}$ e todas as restantes cláusulas são verdadeiras em $\mathcal{I}$.

3.4.17. DEFINIÇÃO. Sejam $\mathcal{C}$ um conjunto de cláusulas e $\mathcal{I}$ uma interpretação para $\mathcal{C}$. Sejam C_1 e C_2 cláusulas na assinatura de $\mathcal{C}$ tal que pelo menos uma de C_1, C_2 é falsa em $\mathcal{I}$. Então diz-se de todas as resolventes de C_1 e C_2 que são *resolventes $\mathcal{I}$ semânticas.* Considere-se agora o conjunto $\mathcal{C} \cup \mathcal{D}$; seja $\mathcal{C}$ o conjunto de cláusulas verdadeiras em $\mathcal{I}$ e $\mathcal{D}$ o conjunto de cláusulas falsas em $\mathcal{I}$. Seja

$$\rho_{\mathcal{I}}(\mathcal{D}) = \{E \mid E \text{ é uma resolvente } \mathcal{I} \text{ semântica de } \mathcal{D}\}.$$

Definimos o operador da resolução semântica $R_{\mathcal{I}}$ como

$$R_{\mathcal{I}}(\mathcal{D}) = \mathcal{D} \cup \rho_{\mathcal{I}}(\mathcal{D}).$$

3.4.18. DEFINIÇÃO. Dado um conjunto de cláusulas $\mathcal{C}$, sejam $\mathcal{I}$ uma interpretação e $\mathcal{P}$ um ordenamento de $P, Q, ... \in Pred(\mathcal{C})$. Um conjunto finito de cláusulas de $\mathcal{C}$ constitui uma sequência de *choque semântico* (ou *choque PI*) $\Xi_{\mathcal{PI}} = (N; E_1, ..., E_n)$ em que N denota o núcleo e $E_1, ..., E_n$ os eletrões ou satélites de $\Xi(\mathcal{C})$ sse

1. $E_1, ..., E_n$ são falsos em $\mathcal{I}$;
2. Seja $C_1 = N$. Para cada $i = 1, ..., n$ existe uma resolvente C_{i+1} de C_i e E_i;
3. O literal resolvido de E_i contém o maior símbolo de predicado em E_i, $i = 1, ..., n$;
4. C_{n+1} é falsa em $\mathcal{I}$.[5]

C_{n+1} chama-se a resolvente PI de $\Xi_{\mathcal{PI}}(\mathcal{C}) = (N; E_1, ..., E_n)$.

[5]Lembramos que $\square$ é sempre falsa em qualquer interpretação (prop. 1.1.32).

3.4.19. EXEMPLO. Sejam $E_1 = P \vee R$, $E_2 = Q \vee R$ e $N = \neg P \vee \neg Q \vee R$. Dados a interpretação $\mathcal{I} = \{\neg P, \neg Q, \neg R\}$ e o ordenamento $P > Q > R$, então $(N; E_1, E_2)$ é um choque PI e a resolvente deste choque é R. Note-se que R é falsa em $\mathcal{I}$.

Além desta restrição dos sinais da sequência de choque, a hiper-resolução impõe outras restrições no espaço de escolha das cláusulas a resolver, evitando assim que muitas fórmulas redundantes sejam produzidas e adicionadas ao espaço da procura. Demonstramos apenas a completude da resolução semântica, da qual a hiper-resolução é um tipo, uma vez que abaixo ela ser-nos-á útil (secção 4.3). (Leitsch, 1997, p. 136ss dá uma demonstração da completude da hiper-resolução.) Dado que utilizando uma qualquer interpretação $\mathcal{I}$ e um qualquer ordenamento $\mathcal{P}$ podemos sempre obter uma dedução PI de $\square$ a partir de um conjunto de cláusulas básicas insatisfazível, demonstramos a completude da resolução semântica pela demonstração da completude da resolução PI.

3.4.20. TEOREMA (completude da resolução PI). Se $\mathcal{P}$ é um ordenamento de símbolos de predicados num conjunto finito e insatisfazível de cláusulas $\mathcal{C}$ e se $\mathcal{I}$ é uma interpretação de $\mathcal{C}$, então existe uma dedução PI de $\square$ a partir de $\mathcal{C}$.

Demonstração. A demonstração é por indução com respeito ao número de átomos de $\mathcal{C}$. Seja $\mathcal{C}$ um conjunto insatisfazível de cláusulas básicas. Seja que $A(\mathcal{C}) = \{P\}$. Então, $\mathcal{C} = \{P, \neg P\} = \mathcal{C}'$. Obviamente, a resolvente de $\mathcal{C}'$ é $\square$, seja qual for a interpretação $\mathcal{I}$ (i.e., $\mathcal{I} = \{P\}$ ou $\mathcal{I} = \{\neg P\}$). Logo, P ou $\neg P$ é falso em $\mathcal{I}$ e $\square$ é de igual modo falsa em $\mathcal{I}$, e temos que $\square$ é uma resolvente PI.

Demonstrámos que o teorema vale para $n = 1$. Suponhamos agora que o teorema vale para $|A(\mathcal{C})| = i, 1 \leq i \leq n$. Para completar a indução, consideramos que $|A(\mathcal{C})| = n + 1$. Começamos por procurar uma cláusula unitária L falsa em $\mathcal{I}$.

(i) $\mathcal{C}$ contém uma cláusula unitária L que é falsa em $\mathcal{I}$. Então, pela eliminação das cláusulas em $\mathcal{C}$ contendo L e eliminando $\neg L$ nas restantes cláusulas de $\mathcal{C}$ obtemos $\mathcal{C}'$. Obviamente, $\mathcal{C}'$ é insatisfazível (cf. regra do literal único; exemplo 3.3.1). $\mathcal{C}'$ contém n ou menos do que n átomos, pelo que pela hipótese da indução existe uma dedução PI de $\square$ a partir de $\mathcal{C}'$. Denotemos esta dedução PI por D'; podemos obter uma dedução de $\square$ de $\mathcal{C}$ a partir de D' bastando para tal substituir cada sequência de choque $\left(N'; E_1', ..., E_q'\right)$, em que $N', E_1', ..., E_q$ são cláusulas ligadas aos nós iniciais de D' e N' se obteve de N por eliminação de $\neg L$, pela sequência de choque PI $\left(N, L; E_1', ..., E_q'\right)$. Se E_i' se obteve de E_i pela eliminação nela de $\neg L$, junta-se o choque PI $(L; E_i)$ acima do nó de E_i'. Obtemos assim uma dedução PI de $\square$ a partir de $\mathcal{C}$.

(ii) $\mathcal{C}$ não contém uma cláusula unitária L falsa em $\mathcal{I}$. Neste caso, podemos obter uma dedução PI D' de $\square$ a partir de $\mathcal{C}'$, em que $\mathcal{C}'$ se obtém pela aplicação da regra do literal único a um literal L que é o símbolo de predicado menor num qualquer conjunto $\{B, \neg B\} \subseteq A(\mathcal{C})$ e é falso em $\mathcal{I}$. Obviamente, $\mathcal{C}'$ é insatisfazível. $\mathcal{C}'$ contém n ou menos do que n átomos, pelo que pela hipótese da indução existe uma dedução PI D' de $\square$ a partir de $\mathcal{C}'$. Coloque-se agora de novo o literal L nas cláusulas das quais primeiramente foi removido e denote-se por D_1 a dedução obtida de D' por esta operação: D_1 continua a ser uma dedução PI, uma vez que L contém o símbolo de predicado menor e é falso em $\mathcal{I}$. É evidente que $D_1 = \square$ ou $D_1 = L$. No primeiro caso, a demonstração está concluída. No segundo caso, por (i) obtemos uma dedução PI $D_2 = \square$ a partir de $\mathcal{C} \cup \{L\}$, em que L é uma cláusula unitária e é falso em $\mathcal{I}$. Pela combinação de D_1 e D_2 obtém-se uma dedução PI de $\square$ a partir de $\mathcal{C}$. $\square$

O teorema de Herbrand (versão II; teorema 3.2.6) e o *lifting lemma* (teorema 3.3.22) garantem que este resultado (a completude da resolução PI) vale para um qualquer conjunto de cláusulas insatisfazível, ou seja, um conjunto de cláusulas não básicas (cf. Chang & Lee, 1973, p. 107).

3.5 A resolução como procedimento de demonstração

3.5.1 A resolução e os problemas SAT e da decisão

O problema SAT (def. 3.1.1) tem múltiplas aplicações, de entre as quais uma das mais importantes é a demonstração automática de teoremas. Como sabemos já, os problemas SAT e da decisão são recursivamente equivalentes, ou seja, uma fórmula ϕ é válida sse $\neg\phi$ é insatisfazível. Assim sendo, podemos testar a validade de um teorema (um conjunto de cláusulas) pela testagem da sua insatisfazibilidade. A aplicação de um demonstrador de teoremas por resolução completo Ψ a um conjunto de cláusulas $\mathcal{C}$ produz um dos três resultados seguintes (cf. secção 3.1.1):

(i) Derivação de $\square$.

(ii) $\Psi(\mathcal{C})$ é finito (ou seja, dado $\mathcal{C}$ verifica-se que Ψ termina) e $\Psi(\mathcal{C})$ não contém uma refutação. Visto que Ψ é completo, sabemos que $\mathcal{C}$ é satisfazível.

(iii) $\Psi(\mathcal{C})$ é infinito e não contém uma refutação de $\mathcal{C}$. Dado $\mathcal{C}$, verifica-se que Ψ não termina; uma vez que Ψ é completo, $\mathcal{C}$ é satisfazível.

Correspondentemente, um refinamento de resolução Res_{rf} é um método de demonstração completo na medida em que, dado um conjunto de cláusulas $\mathcal{C}$ como *input*, há três possibilidades :

(i*) Res_{rf} termina e refuta $\mathcal{C}$, i.e., produz $\square$. Sabemos que $\mathcal{C}$ é insatisfazível.

(ii*) Res_{rf} termina sem produzir $\square$. Dado que Res_{rf} é completo, $\mathcal{C}$ deve ser satisfazível.

(iii*) Res_{rf} não termina. $Res_{rf}(\mathcal{C})$ é infinito e $\square \notin Res_{rf}(\mathcal{C})$. $\mathcal{C}$ deve ser satisfazível, mas $Res_{rf}(\mathcal{C})$ não nos permite detetar esta propriedade.

3.5.2 Implementações da resolução no Prover9/Mace4

Dependendo do *input*, a tarefa de demonstração de um teorema pode não ser trivial, podendo mesmo ser basicamente impossível para um computador humano. Como se disse acima, o grupo no *Argonne National Laboratory* desenvolveu um demonstrador para LPOC baseado na refutação, Otter, o qual funcionava implementando procedimentos de resolução. O seu sucessor atual é conhecido como Prover9/Mace4 e implementa a resolução binária. (O Prover9 implementa ainda procedimentos de demonstração para a lógica equacional, i.e., paramodulação.) O Prover9 é altamente eficiente na demonstração de teoremas por resolução, uma vez que além de fazer as transformações necessárias para lógica clausal, efetua ainda um algoritmo de unificação, sendo assim completamente autónomo no que diz respeito a este procedimento de demonstração. Podemos assim deixar a tarefa da demonstração completamente a cargo do Prover9 em casos mais complexos, ou recorrer a ele apenas para partes mais complicadas de uma demonstração. (Note-se que o Prover9/Mace4 é um *software* gratuito: **http://www.cs.unm.edu/~mccune/prover9/**.)

3.5.1. Exemplo. Seja dada a seguinte teoria:

$$\begin{array}{ll} (P_1) & \forall x\,(F(x) \rightarrow \exists y\,(G(y) \wedge H(x,y)) \wedge \exists y\,(G(y) \wedge \neg H(x,y))) \\ \underline{(P_2)} & \underline{\exists x\,(J(x) \wedge \forall y\,(G(y) \rightarrow H(x,y)))} \\ (P_3) & \exists x\,(J(x) \wedge \neg F(x)) \end{array}$$

Queremos saber se a teoria é válida. A repetição de átomos em P_1 torna a tarefa da demonstração algo complexa, pelo que o recurso ao Prover9 se mostra aqui útil. Introduzimos então P_1 e P_2 na janela superior e P_3 na janela inferior (ou $\neg P_3$ na janela superior após P_1 e P_2) do Prover9. Obtém-se uma demonstração da validade

da teoria: após 14 passos apenas, e implementando somente a resolução binária, o Prover9 deduz a cláusula vazia (denotada por $).

```
========================== PROOF ============================
% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.06) seconds.
% Length of proof is 14.
% Level of proof is 5.
% Maximum clause weight is 0.
% Given clauses 0.

1 (all x (F(x) -> (exists y (G(y) & H(x,y))) & (exists y (G(y) &
-H(x,y))))) # label(non_clause).  [assumption].
2 (exists x (J(x) & (all y (G(y) -> H(x,y))))) # label(non_clause).
[assumption].
3 (exists x (J(x) & -F(x))) # label(non_clause) # label(goal).
[goal].
4 -J(x) | F(x).  [deny(3)].
7 -F(x) | G(f2(x)).  [clausify(1)].
8 -F(x) | -H(x,f2(x)).  [clausify(1)].
10 J(c1).  [clausify(2)].
12 -J(x) | G(f2(x)).  [resolve(4,b,7,a)].
13 -J(x) | -H(x,f2(x)).  [resolve(4,b,8,a)].
15 -G(x) | H(c1,x).  [clausify(2)].
16 G(f2(c1)).  [resolve(12,a,10,a)].
17 -H(c1,f2(c1)).  [resolve(13,a,10,a)].
19 H(c1,f2(c1)).  [resolve(16,a,15,a)].
20 $F. [resolve(19,a,17,a)].
========================== end of proof ====================
```

O Prover9 demonstrou que $P_1, P_2 \models P_3$ pela insatisfazibilidade de $(P_1 \wedge P_2 \wedge \neg P_3)$ (cf. Secção 1.2.2, nomeadamente os teoremas 1.2.8-9 e 1.2.11), uma vez estas transformadas em FNC.

3.5.2. Exemplo. Aumentamos o grau de complexidade de uma teoria. Mostramos como o Prover9 implementa a resolução na demonstração de um problema proposto por L. Schubert em 1978 conhecido como *Schubert's steamroller* e o qual

pretendia representar um desafio para a demonstração automática de teoremas. O problema é de facto complexo, nomeadamente a nível combinatorial, mas foi resolvido pela primeira vez em 1985 por um cálculo de resolução (cf. Walther, 1985). Apresentamos o problema formulado em inglês e a nossa respetiva tradução para LPOC:

> *Wolves, foxes, birds, caterpillars, and snails are animals, and there are some of each of them. Also there are some grains, and grains are plants. Every animal either likes to eat all plants or all animals much smaller than itself that like to eat some plants. Caterpillars and snails are much smaller than birds, which are much smaller than foxes, which are in turn much smaller than wolves. Wolves do not like to eat foxes or grains, while birds like to eat caterpillars but not snails. Caterpillars and snails like to eat some plants. Prove there is an animal that likes to eat a grain-eating animal.*

(P_1) $\forall x\,(W\,(x) \rightarrow A\,(x)) \wedge \exists x\,(W\,(x))$

(P_2) $\forall x\,(F\,(x) \rightarrow A\,(x)) \wedge \exists x\,(F\,(x))$

(P_3) $\forall x\,(B\,(x) \rightarrow A\,(x)) \wedge \exists x\,(B\,(x))$

(P_4) $\forall x\,(C\,(x) \rightarrow A\,(x)) \wedge \exists x\,(C\,(x))$

(P_5) $\forall x\,(S\,(x) \rightarrow A\,(x)) \wedge \exists x\,(S\,(x))$

(P_6) $\exists x\,(G\,(x)) \wedge \forall x\,(G\,(x) \rightarrow P\,(x))$

(P_7) $\forall x(A\,(x) \rightarrow (\forall y\,(P\,(y) \rightarrow Eats\,(x,y)) \vee$
$\forall y\,((A\,(y) \wedge Smaller\,(y,x) \wedge \exists z\,(P\,(z) \wedge Eats\,(y,z))) \rightarrow Eats\,(x,y))))$

(P_8) $\forall x \forall y\,(C\,(x) \wedge B\,(y) \rightarrow Smaller\,(x,y))$

(P_9) $\forall x \forall y\,(S\,(x) \wedge B\,(y) \rightarrow Smaller\,(x,y))$

(P_{10}) $\forall x \forall y\,(B\,(x) \wedge F\,(y) \rightarrow Smaller\,(x,y))$

(P_{11}) $\forall x \forall y\,(F\,(x) \wedge W\,(y) \rightarrow Smaller\,(x,y))$

(P_{12}) $\forall x \forall y\,(W\,(x) \wedge (F\,(y) \vee G\,(y)) \rightarrow \neg Eats\,(x,y))$

(P_{13}) $\forall x \forall y\,(B\,(x) \wedge C\,(y) \rightarrow Eats\,(x,y))$

(P_{14}) $\forall x \forall y\,(B\,(x) \wedge S\,(y) \rightarrow \neg Eats\,(x,y))$

(P_{15}) $\forall x\,(C\,(x) \rightarrow \exists y\,(P\,(y) \wedge Eats\,(x,y)))$

(P_{16}) $\underline{\forall x\,(S\,(x) \rightarrow \exists y\,(P\,(y) \wedge Eats\,(x,y)))}$

(P_{17}) $\exists x \exists y\,(A\,(x) \wedge (A\,(y) \wedge \exists z\,(G\,(z) \wedge Eats\,(y,z))) \wedge Eats\,(x,y))$

Segue-se a demonstração pelo Prover9.

```
========================== PROOF ============================
% -------- Comments from original proof --------
% Proof 1 at 0.01 (+ 0.06) seconds.
% Length of proof is 65.
% Level of proof is 9.
% Maximum clause weight is 13.
% Given clauses 31.

1 (all x (W(x) -> A(x))) & (exists x W(x)) # label(non_clause).
[assumption].
2 (all x (F(x) -> A(x))) & (exists x F(x)) # label(non_clause).
[assumption].
3 (all x (B(x) -> A(x))) & (exists x B(x)) # label(non_clause).
[assumption].
5 (all x (S(x) -> A(x))) & (exists x S(x)) # label(non_clause).
[assumption].
6 (exists x G(x)) & (all x (G(x) -> P(x))) # label(non_clause).
[assumption].
7 (all x (A(x) -> (all y (P(y) -> Eats(x,y))) | (all y (A(y) &
Smaller(y,x) & (exists z (P(z) & Eats(y,z))) -> Eats(x,y)))))
# label(non_clause).  [assumption].
9 (all x all y (S(x) & B(y) -> Smaller(x,y))) # label(non_clause).
[assumption].
10 (all x all y (B(x) & F(y) -> Smaller(x,y))) # label(non_clause).
[assumption].
11 (all x all y (F(x) & W(y) -> Smaller(x,y))) # label(non_clause).
[assumption].
12 (all x all y (W(x) & (F(y) | G(y)) -> -Eats(x,y)))
# label(non_clause).  [assumption].
14 (all x all y (B(x) & S(y) -> -Eats(x,y))) # label(non_clause).
[assumption].
16 (all x (S(x) -> (exists y (P(y) & Eats(x,y))))) # label(non_clause).

[assumption].
17 (exists x exists y (A(x) & A(y) & (exists z (G(z) & Eats(y,z))) &

Eats(x,y))) # label(non_clause) # label(goal).  [goal].
```

```
18 W(c1).  [clausify(1)].
19 -W(x) | A(x).  [clausify(1)].
20 -F(x) | -W(y) | Smaller(x,y).  [clausify(11)].
21 -W(x) | -F(y) | -Eats(x,y).  [clausify(12)].
22 -W(x) | -G(y) | -Eats(x,y).  [clausify(12)].
23 F(c2).  [clausify(2)].
24 -F(x) | A(x).  [clausify(2)].
25 -B(x) | -F(y) | Smaller(x,y).  [clausify(10)].
26 -F(x) | Smaller(x,c1).  [resolve(20,b,18,a)].
27 -F(x) | -Eats(c1,x).  [resolve(21,a,18,a)].
28 B(c3).  [clausify(3)].
29 -B(x) | A(x).  [clausify(3)].
31 -S(x) | -B(y) | Smaller(x,y).  [clausify(9)].
33 -B(x) | -S(y) | -Eats(x,y).  [clausify(14)].
34 -B(x) | Smaller(x,c2).  [resolve(25,b,23,a)].
41 S(c5).  [clausify(5)].
42 -S(x) | A(x).  [clausify(5)].
43 -S(x) | P(f2(x)).  [clausify(16)].
44 -S(x) | Eats(x,f2(x)).  [clausify(16)].
45 -S(x) | Smaller(x,c3).  [resolve(31,b,28,a)].
46 -S(x) | -Eats(c3,x).  [resolve(33,a,28,a)].
47 -G(x) | P(x).  [clausify(6)].
48 G(c6).  [clausify(6)].
49 -A(x) | -A(y) | -G(z) | -Eats(y,z) | -Eats(x,y).  [deny(17)].
50 -G(x) | -Eats(c1,x).  [resolve(22,a,18,a)].
51 Smaller(c2,c1).  [resolve(26,a,23,a)].
52 -A(x) | -P(y) | Eats(x,y) | -A(z) | -Smaller(z,x) |
-P(u) | -Eats(z,u) | Eats(x,z).  [clausify(7)].
53 Smaller(c3,c2).  [resolve(34,a,28,a)].
54 Smaller(c5,c3).  [resolve(45,a,41,a)].
55 A(c1).  [resolve(18,a,19,a)].
56 A(c2).  [resolve(23,a,24,a)].
57 -Eats(c1,c2).  [resolve(27,a,23,a)].
58 A(c3).  [resolve(28,a,29,a)].
63 A(c5).  [resolve(41,a,42,a)].
64 P(f2(c5)).  [resolve(43,a,41,a)].
65 Eats(c5,f2(c5)).  [resolve(44,a,41,a)].
```

```
66 -Eats(c3,c5).  [resolve(46,a,41,a)].
67 P(c6).  [resolve(47,a,48,a)].
68 -A(x) | -A(y) | -Eats(y,c6) | -Eats(x,y).  [resolve(49,c,48,a)].
69 -Eats(c1,c6).  [resolve(50,a,48,a)].
70 -A(c1) | -P(x) | Eats(c1,x) | -A(c2) | -P(y) | -Eats(c2,y) |
Eats(c1,c2).  [resolve(51,a,52,e)].
71 -P(x) | Eats(c1,x) | -P(y) | -Eats(c2,y).
[copy(70),unit_del(a,55),unit_del(d,56),unit_del(g,57)].
72 -A(c2) | -P(x) | Eats(c2,x) | -A(c3) | -P(y) | -Eats(c3,y) |
Eats(c2,c3).  [resolve(53,a,52,e)].
73 -P(x) | Eats(c2,x) | -P(y) | -Eats(c3,y) | Eats(c2,c3).
[copy(72),unit_del(a,56),unit_del(d,58)].
74 -A(c3) | -P(x) | Eats(c3,x) | -A(c5) | -P(y) | -Eats(c5,y) |
Eats(c3,c5).  [resolve(54,a,52,e)].
75 -P(x) | Eats(c3,x) | -P(y) | -Eats(c5,y).
[copy(74),unit_del(a,58),unit_del(d,63),unit_del(g,66)].
79 -P(x) | Eats(c2,x) | -Eats(c3,x) | Eats(c2,c3).  [factor(73,a,c)].

86 -Eats(c2,c6).  [ur(71,a,67,a,b,69,a,c,67,a)].
92 -P(x) | Eats(c3,x).  [resolve(75,d,65,a),unit_del(c,64)].
94 Eats(c3,c6).  [resolve(92,a,67,a)].
98 Eats(c2,c3).  [resolve(94,a,79,c),unit_del(a,67),unit_del(b,86)].

103 $F. [ur(68,a,56,a,b,58,a,c,94,a),unit_del(a,98)].
======================== end of proof ====================
```

Esta demonstração é particularmente interessante, uma vez que apesar da alta complexidade da teoria a demonstração é massiçamente por resolução, apenas com alguns recursos a aplicações de eliminação de unidades (*unit deletion*), uma generalização de primeira ordem da regra do literal único de Davis-Putnam, e de resolução de unidades resultantes (*unit resulting resolution*), que tal como a hiper-resolução permite num único passo a resolução de mais do que uma cláusula-núcleo com outras cláusulas-satélites.

3.6 Exercícios

1. Obtenha o conjunto de cláusulas das seguintes fórmulas:

(a) $\forall y \neg \exists z \left(P\left(y\right) \rightarrow Q\left(y,z\right)\right) \wedge \forall x \forall u \exists w \left(R\left(x,w\right) \leftrightarrow S\left(x,u,w\right)\right)$
(b) $\forall x \left(\left(P\left(x\right) \leftrightarrow Q\left(x\right)\right) \wedge \forall z \neg \exists y \left(R\left(x,z\right) \rightarrow S\left(x,y,z\right)\right)\right)$

2. Para as cláusulas C_1 e C_2, determine se σ é um unificador das mesmas. Determine ainda se σ é um umg de C_1 e C_2.

(a) $C_1 = P\left(a, f\left(y\right), z\right); C_2 = S\left(x, f\left(f\left(b\right)\right), b\right); \sigma = \{x \mapsto a, y \mapsto f\left(b\right), z \rightarrow b\}$
(b) $C_1 = P\left(x, h\left(a,z\right), f\left(x\right)\right); C_2 = P\left(g\left(g\left(v\right)\right), y, f\left(w\right)\right);$
$\sigma = \{x \mapsto g\left(g\left(v\right)\right), y \mapsto h\left(a,z\right), w \mapsto x\}$
(c) $C_1 = Q\left(f\left(x\right), g\left(y\right)\right); C_2 = Q\left(z, g\left(v\right)\right); \sigma = \{x \mapsto a, z \mapsto f\left(a\right), y \mapsto v\}$
(d) $C_1 = conhece\left(joão, x\right); C_2 = conhece\left(joão, joana\right); \sigma = \{x \mapsto joana\}$

3. Determine se os seguintes pares de átomos são unificáveis:

(a) $Q(x,y,z)$ e $Q(u,h(v,v),u)$
(b) $P(x,f(x))$ e $P(y,y)$
(c) $P(f(x))$ e $P(a)$
(d) $P(x,y)$ e $P(y,f(y))$
(e) $S\left(f\left(a\right), g\left(x\right)\right)$ e $S\left(u,u\right)$
(f) $R\left(a,x,h\left(g\left(z\right)\right)\right)$ e $R\left(z,h\left(y\right),h\left(y\right)\right)$

4. Determine as resolventes dos seguintes conjuntos de cláusulas:

(a) $\{\neg P\left(x\right) \vee Q\left(x,b\right), P\left(a\right) \vee Q\left(a,b\right)\}$
(b) $\{\neg P\left(x\right) \vee Q\left(x,y\right), \neg Q\left(a, g\left(a\right)\right)\}$
(c) $\{\neg S\left(x,y,u\right) \vee \neg S\left(y,z.v\right) \vee \neg S\left(x,v,w\right) \vee S\left(u,z,w\right), S\left(h\left(x,y\right),x,y\right)\}$
(d) $\{\neg R\left(v,z,v\right) \vee R\left(w,z,w\right), R\left(a, f\left(b,b\right)\right), a\}$

5. Recorrendo ao demonstrador automático Prover9/Mace4, demonstre a (in)validade das seguintes teorias. Analise em seguida o *output* do demonstrador pormenorizadamente. Para tal, verifique na página do demonstrador alguns processos adicionais àqueles abordados acima (ex.: *merge*, *copy*):
https://www.cs.unm.edu/~mccune/mace4/manual/2009-11A/

(a)
$Q \rightarrow R$

$R \to (P \wedge Q)$
$\underline{P \to (Q \vee R)}$
$P \leftrightarrow Q$

(b)
$\exists x\,(P \to F\,(x))$
$\underline{\exists x\,(F\,(x) \to P)}$
$\exists x\,(P \leftrightarrow F\,(x))$

(c)
$\neg\exists x\,(S\,(x) \wedge Q\,(x))$
$\forall x\,(P\,(x) \to (Q\,(x) \vee R\,(x)))$
$\neg\exists x\,(P\,(x)) \to \exists x\,(Q\,(x))$
$\underline{\forall x\,(Q\,(x) \vee R\,(x) \to S\,(x))}$
$\exists x\,(P\,(x) \wedge R\,(x))$

(d)
$\exists x\,(P\,(x)) \leftrightarrow \exists x\,(Q\,(x))$
$\underline{\forall x \forall y\,(P\,(x) \wedge Q\,(y) \to (R\,(x) \leftrightarrow S\,(y)))}$
$\forall x\,(P\,(x) \to R\,(x)) \leftrightarrow \forall x\,(Q\,(x) \to S\,(x))$

(e)
$\forall z \exists w \forall x \exists y\,[(P\,(x,z) \to P\,(y,w)) \wedge P\,(y,z) \wedge (P\,(y,w) \to \exists u\,(Q\,(u,w)))]$
$\forall x \forall z\,(\neg P\,(x,z) \to \exists y\,(Q\,(y,z)))$
$\underline{\exists x \exists y\,(Q\,(x,y)) \to \forall x\,(R\,(x,x))}$
$\forall x \exists y\,(R\,(x,y))$

4 Resolução assinalada para lógicas multivalentes

4.1 Considerações preliminares

Como se viu acima, a aplicação do princípio de resolução como uma regra de inferência requer que as fórmulas sejam transformadas numa forma clausal normal, nomeadamente em FNCs. Dado que não há formas normais nas lógicas multivalentes, desenvolveram-se estratégias para a obtenção de resultados equivalentes. As tentativas para obter FNCs diretamente nas lógicas multivalentes foram apenas parcialmente satisfatórias, embora seja possível obter equivalência refutacional. Com efeito, tais tentativas são tipicamente ineficazes, ficando-se pelo nível teórico (ex.: Morgan, 1976) ou considerando apenas sistemas lógicos específicos (ex.: Schmitt, 1986). Stachniak (1996, por exemplo) propôs um sistema de demonstração mais geral baseado na resolução para lógicas multivalentes apoiando-se para tal no facto que a operação tarskiana de consequência de LC (def. 1.2.1) pode ser generalizada em grande medida a estas lógicas. No entanto, este sistema de demonstração só é eficazmente aplicável às lógicas funcionalmente completas ou às lógicas de Łukasiewicz, nomeadamente porque o número de verificadores (uma espécie de valores de verdade) pode ser arbitrariamente grande para certas outras classes de lógicas. Por esta e outras razões, o de Stachniak mantém-se um procedimento de resolução em grande medida não mecanizável. Outras tentativas interessantes de implementar o princípio de resolução em lógicas multivalentes, mas agora sobretudo de interesse "histórico" (tal como, aliás, as acima mencionadas), são, por exemplo, as de Di Zenzo (1988), Lee (1972) e Orłowska (1978).

O primeiro aspeto a ter em mente ao tentar implementar o princípio de resolução nas lógicas multivalentes é que estas são *generalizações* de LC. Viu-se acima que a generalização de LC implica propriedades estruturais (normalidade, uniformidade, regularidade K e decisibilidade) que por sua vez se relacionam com propriedades tais como a (quase-)validade, a (quase-)derivabilidade, as (quase-)tautologias e as (quase-)contradições; todas elas permitem "traduzir" lógicas multivalentes para LC. Em

especial, a caracterização algébrica das lógicas multivalentes por meio de matrizes lógicas mostra-se útil na medida em que a seleção dos conjuntos de valores designados se encontra na raiz desta "tradução natural". De facto, o problema SAT para uma qualquer fórmula ϕ num sistema lógico multivalente expressa-se então como "Dá-se alguma vez o caso que ϕ toma um valor de verdade $x \in D$?" Por exemplo, seja S um sistema lógico multivalente para o qual $W = \left\{0, \frac{1}{2}, 1\right\}$ e $D = \{1\}$; a satisfazibilidade da fórmula ϕ pode expressar-se como "É possível valorar ϕ em $\{1\}$?" De igual modo, a insatisfazibilidade de ϕ pode ser assertada se se verifica que ϕ toma sempre um valor em $\left\{0, \frac{1}{2}\right\}$. Temos com efeito que a dualidade clássica entre validade e satisfazibilidade se estende às lógicas multivalentes do seguinte modo: uma fórmula ϕ é D-válida sse não é $\overline{D}$-satisfazível, ou, se definirmos D^+ e D^-, $W = D^+ \cup D^-$, ϕ é D^+-válida sse não é D^--satisfazível.

Ao "marcar" sempre uma fórmula multivalente com o (ou os) valor(es) de verdade que toma ou pode tomar – i.e., o seu *sinal* – obtém-se a *lógica assinalada*. Este formalismo permite a generalização das importantes noções clássicas de validade (invalidade) e satisfazibilidade (insatisfazibilidade) às várias lógicas multivalentes apresentadas acima. Como se sabe, uma valoração em LC indica-se por P e $\neg P$; sabendo-se que $W_{LC} = \{0, 1\}$, podemos *assinalar* (ou seja, atribuir um sinal a) P e $\neg P$ como $\{1\}[P]$ e $\{0\}[P]$, respetivamente. Esta estratégia permite-nos estender o raciocínio clássico bivalente às lógicas multivalentes assinalando-se as fórmulas multivalentes como $S[\phi]$ ou $(W \diagdown S)[\phi]$, para $S \subseteq W$. Ao permitir ainda a construção de FNCs, a *lógica clausal assinalada* permite a aplicação direta da resolução às lógicas multivalentes.

Em seguida damos as definições e resultados básicos que nos permitirão a exploração da implementação do princípio de resolução nas lógicas multivalentes. Começamos por expôr conceitos importantes da lógica assinalada, a qual já introduzimos acima muito sumariamente. Chamamos a atenção para o facto que ao trabalharmos em lógica assinalada evitamos distinções tecnicamente difíceis e frequentemente vagas entre metalinguagens e linguagens-objeto. Ao associarmos simplesmente uma ordem única a um conjunto de valores de verdade dito multivalente podemos, sendo dada uma qualquer lógica multivalente Λ, introduzir uma nova lógica Λ_S cujos átomos são fórmulas assinaladas, podendo então trabalhar diretamente nesta lógica. Isto estende-se sem problemas à LPO e muitas técnicas de inferência podem ser naturalmente generalizadas a tais lógicas (Murray & Rosenthal, 1993). Isto implica que podemos aplicar as bem conhecidas técnicas de LC ao estudo das (ainda) não tão bem conhecidas lógicas multivalentes. (Pode-se, é claro, ainda argumentar, como o fazem Murray & Rosenthal (1993), que todo o raciocínio humano é "essencialmente"

clássico; ver acima, Secção 2.2.) Contudo, deve-se notar que o raciocínio baseado em cláusulas aplicado às lógicas multivalentes é independente da lógica da qual se obtiveram as formas clausais. Neste contexto, a distinção de Baaz et al. (2001) entre procedimentos de demonstração internos e externos pode ser útil; segundo estes autores, a lógica assinalada é um procedimento de demonstração externo com respeito às lógicas multivalentes.

4.2 Lógica assinalada

4.2.1 Definições básicas

Em *lógica assinalada*, a cada fórmula ϕ numa lógica multivalente associa-se um conjunto $S \subseteq W$ (ou um valor $v_i \in W$) e considera-se então o par $(S, \phi)((v_i, \phi)$, respetivamente), como um átomo da álgebra booleana. A lógica assinalada permite-nos criar fórmulas assinaladas que podem ser usadas para traduzir uma qualquer fórmula finitamente multivalente numa fórmula assinalada em FNC equisatisfazível (as fórmulas assinaladas em FNC são uma subclasse de fórmulas assinaladas). É importante realçar que a lógica assinalada é em grande medida comparável a LC em termos de complexidade computacional (cf. Beckert et al., 2000), constituindo assim um "compromisso" interessante entre a expressividade e a complexidade, um tópico explorado em Caleiro & Marcos (2009). Em especial, toda e qualquer fórmula de uma qualquer lógica finitamente multivalente pode ser traduzida para uma equisatisfazível fórmula assinalada em FNC em tempo polinomial, reduzindo assim polinomialmente o problema SAT nas lógicas finitamente multivalentes ao *problema SAT assinalado*.

4.2.1. DEFINIÇÃO. Seja S um subconjunto de um conjunto de valores de verdade W_n, $n \geq 2$, e seja ϕ uma fórmula proposicional de uma lógica multivalente. Uma expressão da forma $S\,[\phi]$, em que S é o *sinal* de ϕ, chama-se uma *fórmula assinalada*.[1]

Representamos uma fórmula assinalada por um valor isolado como $v_i\,[\phi]$ e lemos "ϕ toma o valor v_i". Para $v_1, ..., v_n \in S$, S assinala a fórmula ϕ, escrevemos $\{v_1, ..., v_n\}\,[\phi]$ e lemos "ϕ toma um dos valores v_i no conjunto $S = \{v_1, ..., v_n\}$", respetivamente.

[1] A consideração de conjuntos de valores de verdade (vs. valores isolados) acelera exponencialmente a demonstração, pelo que é comum conceber o sinal de uma fórmula como sendo sempre $S \subseteq W$.

4.2.2. DEFINIÇÃO. Seja P um átomo. Uma expressão da forma $S\,[P]$, em que S é o sinal de P, chama-se um *literal assinalado*. O *complemento* do literal assinalado $L = S\,[P]$, denotado por $\overline{L} = \overline{S}\,[P]$, é $(W \diagdown S)\,[P]$.

4.2.3. DEFINIÇÃO (subsunção). Diz-se que um literal assinalado $S\,[P]$ *subsume* um literal assinalado $S'\,[P']$, denotado por $S\,[P] \subseteq S'\,[P']$, sse $P = P'$ e $S \subseteq S'$.

4.2.4. DEFINIÇÃO. A fórmula assinalada $S\,[\phi]$ é uma *expressão de fórmula assinalada* (EFA) com sinal S. $\blacksquare$ (ou $\top$) e $\square$ (ou $\bot$) são EFAs. Se ϕ e ψ são EFAs, então $(\phi \wedge \psi)$, $(\phi \vee \psi)$, $(\phi \rightarrow \psi)$, $(\phi \leftrightarrow \psi)$ e $\neg\phi$ são EFAs. Se x é uma variável e ϕ é uma EFA, então $(\forall x)\,[\phi]$ and $(\exists x)\,[\phi]$ são EFAs. A relação de satisfação $\models$ entre uma interpretação $\mathcal{I}$ e EFAs define-se indutivamente como no exemplo 1.2.6, bastando para tal acrescentar a condição

- $\mathcal{I} \models S\,[\phi]$ sse $val_{\mathcal{I}}\,(\phi) \in S$.

Obtemos assim a importante propriedade da lógica assinalada de acordo com a qual o comportamento dos conetores e dos quantificadores é absolutamente clássico.

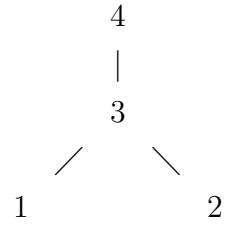
4.2.2 Lógica clausal assinalada

4.2.5. DEFINIÇÃO. Uma fórmula assinalada é uma conjunção de cláusulas assinaladas. Uma *cláusula assinalada* é um conjunto finito de literais assinalados. Uma cláusula assinalada chama-se uma *cláusula unitária assinalada* se contém exatamente um literal e uma *cláusula binária assinalada* se contém exatamente dois literais. Dizemos que uma cláusula assinalada é *vazia* e denotamo-la por $\square$ se todos os seus literais têm o sinal $\{\}$.

4.2.6. DEFINIÇÃO. Uma *fórmula em FNC assinalada* (FNCA) é um conjunto finito de cláusulas assinaladas. Uma fórmula FNCA constituída por cláusulas binárias é uma *fórmula em FNC-2 assinalada*. Uma fórmula assinalada cujos sinais dos literais são da forma $\{v_j \in W \mid j \geq i\}$, denotada por $\uparrow v_i$, ou da forma $\{v_j \in W \mid j \leq i\}$, denotada por $\downarrow v_i$, para $v_i, v_j \in W$ e $\leq$ uma ordem (parcial) em W, chama-se uma *fórmula em FNC regular* (FNCR). Uma fórmula assinalada cujos sinais são conjuntos unitários é uma *fórmula em FNC monoassinalada* (FNCM). (Abreviamos $\uparrow v_i$ e $\downarrow v_i$ como $\uparrow i$ e $\downarrow i$, respetivamente.)

4.2.7. EXEMPLO. Seja $W = \{1, 2, 3, 4\}$ e seja que W é parcialmente ordenado

como



Então, tem-se que $\uparrow 1 = \{1, 3, 4\}$, $\downarrow 1 = \{1\}$, $\overline{\uparrow 1} = \{2\}$ e $\overline{\uparrow 3} = \{1, 2\}$. Os dois primeiros sinais são regulares e os dois últimos são complementos de sinais regulares, enquanto os sinais $\{3\}$ e $\{1, 4\}$ não são de nenhum dos dois tipos, dado que não há nenhum i em W tal que $\uparrow i = \{3\}$ ou $\uparrow i = \{1, 4\}$, $\downarrow i = \{3\}$ ou $\downarrow i = \{1, 4\}$, tal como também não há nenhum i em W tal que $\overline{\uparrow i} = \{3\}$, etc. Note-se ainda que $\overline{\uparrow 3}$ não é regular, embora seja o complemento de um sinal regular.

Quando se utilizam cláusulas assinaladas e/ou fórmulas FNCAs, a regularidade (tal como é definida abaixo) permite a generalização direta da noção de cláusulas de Horn (def. 1.1.34); podemos falar de cláusulas de Horn regulares e de fórmulas de Horn regulares ao atribuirmos a um sinal regular da forma $\uparrow i$ uma *polaridade positiva*, e a um sinal regular da forma $\downarrow i$ uma *polaridade negativa*.

4.2.8. Definição. Uma cláusula regular é uma *cláusula de Horn regular* sse contém no máximo um literal de polaridade positiva e os sinais de todos os seus outros literais são complementos de sinais com polaridade positiva. Uma fórmula FNCR é uma *fórmula de Horn regular* sse todas as suas cláusulas são cláusulas de Horn regulares.

4.2.9. Exemplo. Dado o conjunto W do exemplo 4.2.7 e a ordem a ele associada, o seguinte é um conjunto de cláusulas de Horn regulares: $\left\{\uparrow 1\,[P], \uparrow 2\,[P] \vee \overline{\uparrow 3}\,[Q], \uparrow 4\,[Q]\right\}$. A cláusula $\downarrow 1\,[P]$ é uma cláusula de Horn regular, visto que $\downarrow 1 = \overline{\uparrow 2}$, enquanto $\downarrow 4\,[P]$ não é uma cláusula de Horn regular ($\downarrow 4 \neq \overline{\uparrow i}$).

4.2.3 Satisfazibilidade em lógica assinalada

4.2.10. Definição. Um literal assinalado $S\,[P]$ *é satisfeito* exatamente pelas interpretações $\mathcal{I}$ tais que $val_{\mathcal{I}}(P) \in S$. Uma interpretação *satisfaz* uma cláusula assinalada sse satisfaz pelo menos um dos seus literais assinalados e satisfaz uma fórmula FNCA sse satisfaz todas as suas cláusulas. Uma fórmula FNCA *é satisfazível* sse existe pelo menos uma interpretação que satisfaz todas as suas cláusulas assinaladas; caso contrário, é *insatisfazível*. A cláusula assinalada vazia é sempre

insatisfazível e a fórmula FNCA vazia é sempre satisfazível.

É pois evidente que a semântica das fórmulas FNCAs é clássica *acima do nível do literal*, razão pela qual os procedimentos de demonstração clássicos podem ser naturalmente generalizados às mesmas, bastando para tal um cuidado especial ao nível do literal.

4.2.11. DEFINIÇÃO. Duas cláusulas assinaladas (duas fórmulas FNCAs) são *equivalentes* sse são satisfeitas pelas mesmas interpretações. Diz-se que são *equisatisfazíveis* sse são ambas satisfazíveis ou são ambas insatisfazíveis.

4.2.12. PROPOSIÇÃO. Seja A a fórmula assinalada $A = (\uparrow i_1 [P_1] \wedge ... \wedge \uparrow i_k [P_k]) \rightarrow \uparrow j [Q]$. Então, pelo teorema da dedução (teorema 1.2.9) e pelo teorema 1.2.11, uma interpretação $\mathcal{I}$ satisfaz A sse não satisfaz um dos $\uparrow i_1 [P_1] \wedge ... \wedge \uparrow i_k [P_k]$ ou satisfaz $\uparrow j [Q]$. Logo, a fórmula assinalada A é equivalente à fórmula de Horn assinalada $B = \overline{\uparrow i_1} [P_1] \vee ... \vee \overline{\uparrow i_k} [P_k] \vee \uparrow j [Q]$.

Esta equivalência mostra como a lógica assinalada generaliza de modo natural a LPC; com efeito, pelo teorema 1.2.11, $\uparrow j [Q]$ é uma consequência lógica de $(\uparrow i_1 [P_1] \wedge ... \wedge \uparrow i_k [P_k])$ sse a negação de $(\uparrow i_1 [P_1] \wedge ... \wedge \uparrow i_k [P_k]) \rightarrow \uparrow j [Q]$ é insatisfazível, i.e., sse

$$(\uparrow i_1 [P_1] \wedge ... \wedge \uparrow i_k [P_k]) \not\models \uparrow j [Q] .$$

Pela definição clássica da implicação material, a qual é imediatamente generalizável às lógicas multivalentes, temos

$$\overline{(\uparrow i_1 [P_1] \wedge ... \wedge \uparrow i_k [P_k])} \vee \uparrow j [Q] = B.$$

De igual modo, aplicando a negação a esta fórmula, temos o resultado

$$\overline{\overline{(\uparrow i_1 [P_1] \wedge ... \wedge \uparrow i_k [P_k])} \vee \uparrow j [Q]} \equiv \uparrow i_1 [P_1] \wedge ... \wedge \uparrow i_k [P_k] \wedge \overline{\uparrow j} [Q] .$$

4.2.13. PROPOSIÇÃO. Para todos os sinais $S_1, ..., S_n \subseteq W, n \in \mathbb{N}$ e toda a variável proposicional P,

$$S_1 [P] \vee ... \vee S_n [P] \vee D \equiv (S_1 \cup ... \cup S_n) [P] \vee D.$$

Esta proposição deve ser aplicada com cuidado, uma vez que é verdadeira sse não

destrói a regularidade de uma cláusula.

4.2.4 Transformação/tradução para forma clausal

4.2.4.1 Noções gerais

Acima mencionámos a vantagem que a lógica assinalada apresenta com respeito ao SAT nas lógicas finitamente multivalentes. Com efeito, não só toda e qualquer fórmula finitamente multivalorada pode ser traduzida para uma fórmula FNCA equisatisfazível em tempo polinomial, mas toda e qualquer fórmula FNCA pode ser traduzida para uma fórmula FNCR equisatisfazível com uma qualquer ordem total em W também em tempo polinomial. Beckert et al. (2000) apresentam um procedimento simples em dois passos:

(1) Primeiramente, pela proposição 4.2.13, uma cláusula assinalada contendo literais da forma $S\,[P]$ é transformada numa cláusula monoassinalada pela substituição de $S\,[P]$ por $\bigvee_{i\in S}\{i\}\,[P]$.

(2) Procede-se então à eliminação de todas as ocorrências de literais monoassinalados pela substituição de uma cláusula $C = \{i\}\,[P] \vee D$ por

$$C_1 = \uparrow i\,[P] \vee S\,[Q],\ C_2 = \downarrow i\,[P] \vee S\,[Q] \text{ e } C_3 = D \vee \overline{S}\,[Q],$$

em que S é um qualquer sinal regular e na condição de a nova variável Q não ocorrer em nenhum outro lugar.

Um senão óbvio deste método é a multiplicação de cláusulas no processo de tradução, mas pelo menos em alguns casos tal pode ser evitado ou melhorado, dependendo do tipo de ordem em W (e.g., Sofronie-Stokkermans, 1998).

Baaz et al. (2001) apresentam um procedimento de transformação para fórmulas assinaladas que permite uma formalização da semântica multivalente; por seu lado, isto permite que se raciocine classicamente nestas lógicas, nomeadamente por meio das seguintes equivalências:

4.2.14. Proposição. Para uma qualquer EFA ϕ verificam-se as equivalências seguintes:

(i) $\{\}\,[\phi] \equiv \Box, W\,[\phi] \equiv \blacksquare$
(ii) $\neg\,(S\,[\phi]) \equiv \overline{S}\,[\phi]$

(iii) $S_1\,[\phi] \vee S_2\,[\phi] \equiv (S_1 \cup S_2)\,[\phi]$, em especial $\{v_1\}\,[\phi] \vee ... \vee \{v_n\}\,[\phi] \equiv \{v_1, ..., v_n\}\,[\phi]$

(iv) $S_1\,[\phi] \wedge S_2\,[\phi] \equiv (S_1 \cap S_2)\,[\phi]$, em especial $(\{v_1\}\,[\phi] \wedge \{v_2\}\,[\phi]) \equiv \bot$ para $v_1 \neq v_2$.

Por (ii), as negações podem ser completamente eliminadas das EFAs, e (iii) permite a eliminação de todos os sinais não unitários pela introdução de disjunções. É evidente que a semântica de quaisquer conetores e quantificadores distributivos pode ser expressa por meio de EFAs.

4.2.15. Exemplo. Para algumas lógicas multivalentes (ex.: lógicas de Gödel e de Łukasiewicz), dadas duas fórmulas A e B, a conjunção e a disjunção podem definir-se como

$$A \wedge B = min\,(A, B)$$

e

$$A \vee B = max(A, B)$$

em que *min* e *max* se referem aos valores de verdade de A e B, i.e.,

$$\tilde{\wedge}\,(v_i, v_j) = v_{min(i,j)}$$

e

$$\tilde{\vee}\,(v_i, v_j) = v_{max(i,j)}.$$

Em termos de EFAs, temos então

$$\{v_i\}\,[\phi_1 \wedge \phi_2] \equiv \{v_j \mid j \geq i\}\,[\phi_1] \wedge \{v_j \mid j \geq i\}\,[\phi_2] \wedge (\{v_i\}\,[\phi_1] \vee \{v_i\}\,[\phi_2])$$

e

$$\{v_i\}\,[\phi_1 \vee \phi_2] \equiv \{v_j \mid j \leq i\}\,[\phi_1] \wedge \{v_j \mid j \leq i\}\,[\phi_2] \wedge (\{v_i\}\,[\phi_1] \vee \{v_i\}\,[\phi_2])$$

Então, para uma lógica trivalente com $W = \{\mathrm{V}, \mathrm{I}, \mathrm{F}\}$ e um quantificador universal $\forall^*$ cuja função de verdade para $\mathcal{V} \subseteq W$, $\mathcal{V} \neq \emptyset$, é

$$\widetilde{\forall^*}\,(\mathcal{V}) = \begin{cases} \mathrm{V} & \text{se } \mathcal{V} = \{\mathrm{V}\} \\ \mathrm{F} & \text{se } \mathrm{I} \in \mathcal{V} \text{ ou } \mathrm{F} \in \mathcal{V} \end{cases}$$

a EFA seguinte é obtida como uma especificação da sua semântica:

$$(\{\mathrm{V}\}[(\forall^{*}x)\,A(x)] \equiv (\forall x)\,\{\mathrm{V}\}[A(x)]) \quad \wedge \quad (\{\mathrm{F}\}[(\forall^{*}x)\,A(x)] \equiv (\exists x)\,\{\mathrm{I},\mathrm{F}\}[A(x)])\,.$$

4.2.16. DEFINIÇÃO. Dado um par (ϕ, Φ), em que ϕ é uma fórmula assinalada e Φ é uma EFA, $\phi \Longrightarrow \Phi$ é uma *regra de transformação*. Uma regra de transformação é *correta* sse $\phi \equiv \Phi$ é válida.[2] Uma regra é *redutora* (i.e., é uma *regra de redução*) sse todas as fórmulas multivaloradas em Φ são subfórmulas próprias da fórmula multivalorada em ϕ. Seja $\mathcal{R}$ um conjunto de regras de redução; diz-se que $\mathcal{R}$ é *completo* se para toda a fórmula assinalada $S[\phi]$ existe uma regra de redução em $\mathcal{R}$ que lhe é aplicável, a menos que ϕ seja atómica.

$\mathcal{R}$ é completo se contém regras $S[O(\phi_1, ..., \phi_n)] \Longrightarrow \Phi$ e $S[Qx(\phi)] \Longrightarrow \Phi$ para todo o conetor O (abreviando O_i) de aridade n, todo o quantificador Q (abreviando Q_i) e todo o sinal $S \neq \emptyset$. Obviamente, dado um conjunto completo de regras de redução $\mathcal{R}$ podemos transformar uma qualquer fórmula assinalada numa EFA consistindo apenas em fórmulas assinaladas atómicas. Embora os conjuntos unitários bastem para a completude, a consideração de conjuntos de sinais com cardinalidade maior do que 1 é computacionalmente vantajosa, como já se referiu. Passamos a dar as regras de transformação gerais.

4.2.17. DEFINIÇÃO. Uma *regra de transformação proposicional* é uma expressão da forma

$$S[O(A_1, ..., A_n)] \Longrightarrow \bigwedge_{i \in I} \bigvee_{j \in J} S_{ij}\left[A'_{ij}\right]$$

em que $A'_{ij} \in \{A_1, ..., A_n\}$.

4.2.18. EXEMPLO. Para $W = \{\mathrm{V}, \mathrm{F}\}$ e os conetores $\neg$, $\wedge$ e $\vee$, o seguinte é um conjunto correto de regras de redução:

$$\{\mathrm{V}\}[A \wedge B] \Longrightarrow \{\mathrm{V}\}[A] \wedge \{\mathrm{V}\}[B]$$
$$\{\mathrm{F}\}[A \wedge B] \Longrightarrow \{\mathrm{F}\}[A] \vee \{\mathrm{F}\}[B]$$

[2] A distinção entre uma fórmula assinalada e uma EFA pode ser marcada por uma notação especial para esta última (e.g., Baaz et al., 2001), uma estratégia que se torna supérflua com a definição 4.2.4.

$$\{\mathrm{V}\}\,[\neg A] \Longrightarrow \{\mathrm{F}\}\,[A]$$

$$\{\mathrm{F}\}\,[\neg A] \Longrightarrow \{\mathrm{V}\}\,[A]$$

4.2.19. Exemplo. Dado o conjunto de regras de redução do exemplo 4.2.18, a expressão $\{\mathrm{F}\}\,[A \wedge \neg\,(B \wedge \neg C)]$ passa pelas transformações seguintes:

$$\begin{aligned}
\{\mathrm{F}\}\,[A \wedge \neg\,(B \wedge \neg C)] &\equiv \{\mathrm{F}\}\,[A] \vee \{\mathrm{F}\}\,[\neg\,(B \wedge \neg C)] \\
&\equiv \{\mathrm{F}\}\,[A] \vee \{\mathrm{V}\}\,[B \wedge \neg C] \\
&\equiv \{\mathrm{F}\}\,[A] \vee \{\mathrm{V}\}\,[B] \wedge \{\mathrm{V}\}\,[\neg C] \\
&\equiv \{\mathrm{F}\}\,[A] \vee \{\mathrm{V}\}\,[B] \wedge \{\mathrm{F}\}\,[C]
\end{aligned}$$

4.2.20. Definição. Uma *regra de transformação quantificacional* é uma expressão da forma

$$S\,[(Qx)\,A\,(x)] \Longrightarrow \bigwedge_{i \in I} \left(\bigvee_{j \in J} (\exists x)\, S_{ij}\,[A\,(x)] \vee \bigvee_{k \in K} (\forall x)\, S_{ik}\,[A\,(x)] \right).$$

4.2.21. Exemplo. As regras de redução corretas para o quantificador universal de qL_3, qLG_3, $\mathrm{qL\Pi}_3$ e qP_3 são as seguintes:

$\{\mathrm{V}\}\,[(\forall x)\,A\,(x)] \Longrightarrow (\forall x)\,\{\mathrm{V}\}\,[A\,(x)]$
$\{\mathrm{I}\}\,[(\forall x)\,A\,(x)] \Longrightarrow (\exists x)\,\{\mathrm{I}\}\,[A\,(x)] \wedge (\forall x)\,\{\mathrm{V},\mathrm{I}\}\,[A\,(x)]$
$\{\mathrm{F}\}\,[(\forall x)\,A\,(x)] \Longrightarrow (\exists x)\,\{\mathrm{F}\}\,[A\,(x)]$

4.2.4.2 Regras de transformação para conetores multivalentes

4.2.22. Proposição. Sejam O um conetor de aridade n e i um qualquer valor de verdade. Pela examinação da tabela de verdade de O pode construir-se uma EFA E contendo apenas fórmulas assinaladas a partir do conjunto $\{j\,[A_k] \mid 1 \leq k \leq n, j \in W\}$ e tal que $i\,[O\,(A_1, ..., A_n)] \equiv E$. De facto, podem construir-se duas EFAs (não únicas) C e D tais que

(i) C, D não contêm $\neg$;
(ii) C, D contêm apenas literais do conjunto $\{j\,[A_k] \mid 1 \leq k \leq n, j \in W\}$;
(iii) C está em FNC e D está em FND;

(iv) C, D e E são equivalentes enquanto fórmulas booleanas e são pois equivalentes a $i\left[O\left(A_1, ..., A_n\right)\right]$.

Demonstração. Com efeito, para uma qualquer fórmula A e um qualquer valor de verdade i, temos que

$$(\S) \qquad \neg\left(i\left[A\right]\right) \equiv \bigvee_{j \neq i} j\left[A\right]$$

Logo, dada uma EFA qualquer E pode encontrar-se uma EFA equivalente C que está em FNC e não contém o conetor $\neg$. Mostramos como nas proposições 4.2.23-4. □

4.2.23. Proposição (obtenção de FNDs para conetores multivalentes). Para um qualquer conetor O de aridade n e um qualquer sinal S definimos

$$FND_O\left(S\right) := \bigvee_{\substack{v_1, ..., v_n \in W \\ \tilde{O}\left(v_1, ..., v_n\right) \in S}} \bigwedge_{i=1}^{n} \left\{v_i\right\}\left[A_i\right]$$

$FND_O\left(S\right)$ é maximal no sentido em que toda a FND equivalente a $S\left[O\left(A_1, ..., A_n\right)\right]$ tem no máximo tantos disjuntos como esta, sendo o seu número limitado por $|W|^n$. As formas normais limitadas por $|W|^{n-1}$ têm menos disjuntos e conjuntos. Com efeito, temos

$$fnd_O\left(S\right) := \bigvee_{v_1, ..., v_{n-1} \in W} \left(\bigwedge_{i=1}^{n-1} \left\{v_i\right\}\left[A_i\right] \wedge \left\{ v_n \mid \tilde{O}\left(v_1, ..., v_n\right) \in S \right\}\left[A_n\right] \right)$$

4.2.24. Proposição (obtenção de FNCs para conetores multivalentes). Dadas (§) e as equivalências $\neg FND_O\left(\overline{S}\right) \equiv FNC_O\left(\overline{S}\right)$ ou $\neg fnd_O\left(\overline{S}\right) \equiv fnc_O\left(\overline{S}\right)$ pela proposição 4.2.14 (ii), podemos começar por estas equivalências e aplicar a lei de de Morgan eliminando assim todos os sinais negativos, de modo a obter

$$FNC_O\left(S\right) := \bigwedge_{\substack{v_1, ..., v_n \in W \\ \tilde{O}\left(v_1, ..., v_n\right) \in \bar{S}}} \bigvee_{i=1}^{n} \overline{\left\{v_i\right\}}\left[A_i\right]$$

e

$$fnc_O\left(S\right) := \bigwedge_{v_1, ..., v_{n-1} \in W} \left(\bigvee_{i=1}^{n-1} \overline{\left\{v_i\right\}}\left[A_i\right] \vee \left\{ v_n \mid \tilde{O}\left(v_1, ..., v_n\right) \in S \right\}\left[A_n\right] \right)$$

4.2.25. EXEMPLO. Seja O o conetor $\to$ em Ł$_3$. Para duas fórmulas (atómicas) A e B, queremos computar a FNC de $\{\mathrm{F}\}\,[A \to B]$. Começamos então por computar a FND de $\{\mathrm{V},\mathrm{I}\}\,[A \to B]$, i.e.,

$$\bigvee_{\substack{v_1, v_2 \in \{\mathrm{V},\mathrm{I},\mathrm{F}\} \\ v_1 \to v_2 \neq \mathrm{F}}} (\{v_1\}\,[A] \wedge \{v_2\}\,[B])$$

Por examinação da tabela de verdade obtém-se a FND

$$(\{\mathrm{V}\}\,[A] \wedge \{\mathrm{V}\}\,[B]) \vee (\{\mathrm{V}\}\,[A] \wedge \{\mathrm{I}\}\,[B]) \vee (\{\mathrm{I}\}\,[A] \wedge \{\mathrm{V}\}\,[B]) \vee (\{\mathrm{I}\}\,[A] \wedge \{\mathrm{I}\}\,[B]) \vee$$

$$(\{\mathrm{I}\}\,[A] \wedge \{\mathrm{F}\}\,[B]) \vee (\{\mathrm{F}\}\,[A] \wedge \{\mathrm{V}\}\,[B]) \vee (\{\mathrm{F}\}\,[A] \wedge \{\mathrm{I}\}\,[B]) \vee (\{\mathrm{F}\}\,[A] \wedge \{\mathrm{F}\}\,[B])$$

Uma vez obtida esta FND, a computação da FNC de $\{\mathrm{F}\}\,[A \to B]$ não apresenta quaisquer dificuldades, e obtemos a fórmula

$$(\{\mathrm{I},\mathrm{F}\}\,[A] \vee \{\mathrm{I},\mathrm{F}\}\,[B]) \wedge (\{\mathrm{I},\mathrm{F}\}\,[A] \vee \{\mathrm{V},\mathrm{F}\}\,[B]) \wedge (\{\mathrm{V},\mathrm{F}\}\,[A] \vee \{\mathrm{I},\mathrm{F}\}\,[B]) \wedge$$

$$(\{\mathrm{V},\mathrm{F}\}\,[A] \vee \{\mathrm{V},\mathrm{F}\}\,[B]) \wedge (\{\mathrm{V},\mathrm{F}\}\,[A] \vee \{\mathrm{V},\mathrm{I}\}\,[B]) \wedge (\{\mathrm{V},\mathrm{I}\}\,[A] \vee \{\mathrm{I},\mathrm{F}\}\,[B]) \wedge$$

$$(\{\mathrm{V},\mathrm{I}\}\,[A] \vee \{\mathrm{V},\mathrm{F}\}\,[B]) \wedge (\{\mathrm{V},\mathrm{I}\}\,[A] \vee \{\mathrm{V},\mathrm{I}\}\,[B])$$

Mostramos agora como $fnd_O\,(S)$ e $fnc_O\,(S)$ otimizam a obtenção de formas normais nas lógicas multivalentes pela computação da fnd de $\{\mathrm{V},\mathrm{I}\}\,[A \to B]$ e subsequentemente da fnc de $\{\mathrm{F}\}\,[A \to B]$:

fnd de $\{\mathrm{V},\mathrm{I}\}\,[A \to B] \Longrightarrow$

$$(\{\mathrm{V}\}\,[A] \wedge \{\mathrm{V},\mathrm{I}\}\,[B]) \vee (\{\mathrm{I}\}\,[A] \wedge \{\mathrm{V},\mathrm{I},\mathrm{F}\}\,[B]) \vee (\{\mathrm{F}\}\,[A] \wedge \{\mathrm{V},\mathrm{I},\mathrm{F}\}\,[B]) \equiv$$

$$\equiv (\{\mathrm{V}\}\,[A] \wedge \{\mathrm{V},\mathrm{I}\}\,[B]) \vee \{\mathrm{I}\}\,[A] \vee \{\mathrm{F}\}\,[A]$$

fnc de $\{\mathrm{F}\}\,[A \to B] \Longrightarrow$

$$(\{\mathrm{I},\mathrm{F}\}\,[A] \vee \{\mathrm{F}\}\,[B]) \wedge \{\mathrm{V},\mathrm{F}\}\,[A] \wedge \{\mathrm{V},\mathrm{I}\}\,[A]$$

Fazemos notar as simplificações efetuadas acima: remoção de literais das formas

$\{\}[\phi]$ e $\{\mathrm{V},\mathrm{I},\mathrm{F}\}[\phi]$.

A proposição seguinte captura tudo isto:

4.2.26. PROPOSIÇÃO. Seja O um qualquer conetor de aridade n para n finito. Faça-se ainda E representar uma qualquer (a) $FND_O(S)$, (b) $fnd_O(S)$, (c) $FNC_O(S)$, ou (d) $fnc_O(S)$, em que S é um sinal. Então, para todo O e todo o sinal S, a regra $S[O(A_1,...,A_n)] \Longrightarrow E$ é correta e redutora. O número de disjuntos e conjuntos em (a) e (c) ((b) e (d)) é limitado por $|W|^n$ ($|W|^{n-1}$, respetivamente).

É ainda possível minimizar o número de conjuntos, nomeadamente por meio da subsunção (def. 4.2.3).

4.2.27. EXEMPLO. Para o conetor $\rightarrow$ em Ł$_3$ e aplicando a minimização indicada, obtém-se:

$$\{\mathrm{V}\}[A \rightarrow B] \Longrightarrow$$
$$(\{\mathrm{V}\}[A] \wedge \{\mathrm{V}\}[B]) \vee (\{\mathrm{I}\}[A] \wedge \{\mathrm{V}\}[B]) \vee (\{\mathrm{I}\}[A] \wedge \{\mathrm{I}\}[B]) \vee \{\mathrm{F}\}[A]$$
$$\equiv \{\mathrm{V}\}[B] \vee (\{\mathrm{I}\}[A] \wedge \{\mathrm{I}\}[B]) \vee \{\mathrm{F}\}[A]$$
$$\equiv (\{\mathrm{V}\}[B] \vee \{\mathrm{F}\}[A] \vee \{\mathrm{I}\}[A]) \wedge (\{\mathrm{V}\}[B] \vee \{\mathrm{F}\}[A] \vee \{\mathrm{I}\}[B])$$
$$\equiv (\{\mathrm{F},\mathrm{I}\}[A] \vee \{\mathrm{V}\}[B]) \wedge (\{\mathrm{F}\}[A] \vee \{\mathrm{V},\mathrm{I}\}[B])$$

$$\{\mathrm{I}\}[A \rightarrow B] \Longrightarrow$$
$$(\{\mathrm{V}\}[A] \wedge \{\mathrm{I}\}[B]) \vee (\{\mathrm{I}\}[A] \wedge \{\mathrm{F}\}[B])$$
$$\equiv (\{\mathrm{V}\}[A] \vee \{\mathrm{I}\}[A]) \wedge (\{\mathrm{V}\}[A] \vee \{\mathrm{F}\}[B]) \wedge (\{\mathrm{I}\}[A] \vee \{\mathrm{I}\}[B]) \wedge (\{\mathrm{I}\}[B] \vee \{\mathrm{F}\}[B])$$
$$\equiv \{\mathrm{V},\mathrm{I}\}[A] \wedge (\{\mathrm{V}\}[A] \vee \{\mathrm{F}\}[B]) \wedge (\{\mathrm{I}\}[A] \vee \{\mathrm{I}\}[B]) \wedge \{\mathrm{I},\mathrm{F}\}[B]$$
$$\equiv (\{\mathrm{V}\}[A] \vee \{\mathrm{F}\}[B]) \wedge (\{\mathrm{I}\}[A] \vee \{\mathrm{I}\}[B])$$

$$\{\mathrm{F}\}[A \rightarrow B] \Longrightarrow \{\mathrm{V}\}[A] \wedge \{\mathrm{F}\}[B]$$

4.2.4.3 Regras de transformação para quantificadores multivalentes

Dada a apresentação acima das regras de transformação para conetores multivalentes, as definições seguintes relativas aos quantificadores multivalentes deverão ser evidentes uma vez formuladas algumas definições suplementares.

Recorde-se, da definição 1.1.6, que a *distribuição* de uma fórmula quantificada $Q_i x A(x)$ é o conjunto de valores de verdade que se obtém pela avaliação de $A(x)$ para

todos os valores possíveis de x dado um domínio $\mathscr{D}$, i.e., $distr_{\mathcal{I},x}(\phi) = \left\{val_{\mathcal{I}_x^d}(\phi) \mid d \in \mathscr{D}\right\}$. Por outras palavras, a distribuição de $A(x)$ é a coleção de todos os valores de verdade que se podem obter pela valoração de $A(x)$; assim sendo, $\mathcal{V}$ é a distribuição de $A(x)$ sse $(\forall x)\,\mathcal{V}\,[A(x)]$ e $(\exists x)\,\{v_i\}\,[A(x)]$, i.e., sse o valor atribuído a $A(x)$ se encontra em $\mathcal{V}$ para todo x e sse para todo o valor de verdade $v_i \in \mathcal{V}$ existe um x tal que a valoração de x corresponde a v_i para todo $v_i \in \mathcal{V}$, ou seja, se existe uma função de valoração para os quantificadores de acordo com a qual uma variável pode ser interpretada em W (cf. a função de valoração para um quantificador universal $\forall^*$, i.e., $\widetilde{\forall}^*$, no exemplo 4.2.15).

4.2.28. Proposição (obtenção de FNDs para quantificadores multivalentes). Para um quantificador distributivo Q e um sinal S, temos

$$FND_Q(S) \quad := \bigvee_{\substack{\emptyset \subset \mathcal{V} \subseteq W \\ \tilde{Q}(\mathcal{V}) \in S}} \left((\forall x)\,\mathcal{V}\,[A(x)] \wedge \bigwedge_{v_i \in \mathcal{V}} (\exists x)\,\{v_i\}\,[A(x)] \right)$$

$FND_Q(S)$ tem uma fórmula característica para cada distribuição $\mathcal{V}$ que satisfaz a condição $\tilde{Q}(\mathcal{V})$. O limite para o número de disjuntos em $FND_Q(S)$ é de $2^{|W|}$, mas, tal como no caso das regras de transformação para os conetores multivalentes, este limite pode ser reduzido, nomeadamente para um limite minimal de $2^{|W|-1}$ pela aplicação da definição

$$fnd_{Q,u}(S) \quad := \bigvee_{\mathcal{V} \subseteq (W - \{u\})} \left((\forall x)\,\alpha_S(\mathcal{V})\,[A(x)] \wedge \bigwedge_{v_i \in \beta_S(\mathcal{V})} (\exists x)\,\{v_i\}\,[A(x)] \right)$$

em que u é um valor de verdade qualquer, $u \notin \mathcal{V}$, $\alpha_S(\mathcal{V})$ e $\beta_S(\mathcal{V})$ são dados na tabela abaixo, e $fnd_{Q,u}(S)$ obtém-se pela junção das fórmulas características de $\mathcal{V}$ e $\mathcal{V} \cup \{u\}$ numa única expressão.

$\tilde{Q}(\mathcal{V})$	$\tilde{Q}(\mathcal{V} \cup \{u\})$
$\notin S$	$\notin S$
$\notin S$	$\in S$
$\in S$	$\notin S$
$\in S$	$\in S$

$\alpha_S(\mathcal{V})$	$\beta_S(\mathcal{V})$
$\{\}$	$\{\}$
$\mathcal{V} \cup \{u\}$	$\mathcal{V} \cup \{u\}$
$\mathcal{V}$	$\mathcal{V}$
$\mathcal{V} \cup \{u\}$	$\mathcal{V}$

4.2.29. Proposição (obtenção de FNCs para quantificadores multivalentes). Aplicando a lei de de Morgan a $\neg FND_{Q}\left(\overline{S}\right)$ e $\neg fnd_{Q,u}\left(\overline{S}\right)$ e eliminando todas as negações, obtêm-se as definições

$$FNC_{Q}\left(S\right) \quad := \quad \bigwedge_{\substack{\emptyset \subseteq \mathcal{V} \subseteq W \\ \tilde{Q}\left(\mathcal{V}\right) \in \overline{S}}} \left(\left(\exists x\right) \overline{\mathcal{V}} \left[A\left(x\right)\right] \vee \bigvee_{v_i \in \mathcal{V}} \left(\forall x\right) \overline{\left\{v_i\right\}} \left[A\left(x\right)\right] \right)$$

e

$$fnc_{Q,u}\left(S\right) \quad := \quad \bigwedge_{\mathcal{V} \subseteq (W - \{u\})} \left(\left(\exists x\right) \overline{\alpha_{\overline{S}}\left(\mathcal{V}\right)} \left[A\left(x\right)\right] \vee \bigvee_{v_i \in \beta_{\overline{S}}(\mathcal{V})} \left(\forall x\right) \overline{\left\{v_i\right\}} \left[A\left(x\right)\right] \right).$$

4.2.30. Exemplo. Considere-se o quantificador $\forall$ em qŁ$_3$ definido por $\tilde{\forall}\left(\{\mathrm{V}\}\right) = \mathrm{V}$ e

$$\tilde{\forall}\left(\mathcal{V}\right) = \begin{cases} \mathrm{F} & \text{se } \mathrm{F} \in \mathcal{V} \\ \mathrm{I} & \text{caso contrário} \end{cases}.$$

De modo a construir a $cnf_{\forall,\mathrm{I}}\left\{\mathrm{I}\right\}$, começamos por obter $\alpha_{\overline{\{\mathrm{I}\}}} = \alpha_{\{\mathrm{V,F}\}}$ and $\beta_{\overline{\{\mathrm{I}\}}} = \beta_{\{\mathrm{V,F}\}}$ (cf. as funções de verdade dos quantificadores de qŁ$_3$ no exemplo 2.5.8):

$\mathcal{V}$	$\tilde{\forall}\left(\mathcal{V}\right)$	$\tilde{\forall}\left(\mathcal{V} \cup \{\mathrm{I}\}\right)$
$\{\}$	$-$	I
$\{\mathrm{F}\}$	F	F
$\{\mathrm{V}\}$	V	I
$\{\mathrm{V,F}\}$	F	F

$\alpha_{\{\mathrm{V,F}\}}\left(\mathcal{V}\right)$	$\beta_{\{\mathrm{V,F}\}}\left(\mathcal{V}\right)$
$\{\}$	$\{\}$
$\{\mathrm{I,F}\}$	$\{\mathrm{F}\}$
$\{\mathrm{V}\}$	$\{\mathrm{V}\}$
$\{\mathrm{V,I,F}\}$	$\{\mathrm{V,F}\}$

Computamos em seguida $cnf_{\forall,\mathrm{I}}\left\{\mathrm{I}\right\}$:

$$
\begin{aligned}
cnf_{\forall,\mathrm{I}}\{\mathrm{I}\} \quad = \quad & (\exists x)\,\overline{\{\}}\,[A(x)] \\
& \wedge \Big((\exists x)\,\overline{\{\mathrm{I},\mathrm{F}\}}\,[A(x)] \vee (\forall x)\,\overline{\{\mathrm{F}\}}\,[A(x)]\Big) \\
& \wedge \Big((\exists x)\,\overline{\{\mathrm{V}\}}\,[A(x)] \vee (\forall x)\,\overline{\{\mathrm{V}\}}\,[A(x)]\Big) \\
& \wedge \Big((\exists x)\,\overline{\{\mathrm{V},\mathrm{I},\mathrm{F}\}}\,[A(x)] \vee (\forall x)\,\overline{\{\mathrm{V}\}}\,[A(x)] \vee (\forall x)\,\overline{\{\mathrm{F}\}}\,[A(x)]\Big) \\
\equiv \quad & ((\exists x)\,\{\mathrm{V}\}\,[A(x)] \vee (\forall x)\,\{\mathrm{V},\mathrm{I}\}\,[A(x)]) \\
& \wedge ((\exists x)\,\{\mathrm{I},\mathrm{F}\}\,[A(x)] \vee (\forall x)\,\{\mathrm{I},\mathrm{F}\}\,[A(x)]) \\
& \wedge ((\forall x)\,\{\mathrm{I},\mathrm{F}\}\,[A(x)] \vee (\forall x)\,\{\mathrm{V},\mathrm{I}\}\,[A(x)])
\end{aligned}
$$

Pela aplicação de regras de otimização (ex.: eliminação de fórmulas redundantes), obtém-se a expressão minimal só com dois conjuntos

$$((\exists x)\,\{\mathrm{I}\}\,[A(x)] \wedge (\forall x)\,\{\mathrm{V},\mathrm{I}\}\,[A(x)])\,.$$

Note-se que estas regras de transformação para quantificadores multivalentes são bastante gerais na medida em que as FNC_Q e $fnc_{Q,u}$ são exatamente as mesmas para as lógicas finitamente multivalentes baseadas nas normas-t, bem como para as lógicas de Post finitamente multivalentes, visto que todas elas partilham as mesmas funções de verdade para os quantificadores $\forall$ e $\exists$ (exemplo 2.5.3). Assim sendo, a $cnf_{\forall,\mathrm{I}}\{\mathrm{I}\}$ para o quantificador universal em qL_3 é exatamente a mesma para o mesmo quantificador nos sistemas qLG_3, $\mathrm{qL\Pi}_3$ e qP_3. Efetivamente, embora estes sistemas lógicos difiram grandemente entre si ao nível proposicional (ver acima), são todos semelhantes ao nível quantificado. Logo, é óbvio que as FNDs (fnds) e as FNCs (fncs) como definidas acima são idênticas para qL_n, qLG_n, $\mathrm{qL\Pi}_n$ e qP_n, para um qualquer i, $3 \leq i \leq n$, n finito.

4.2.31. Proposição. Seja Q um qualquer quantificador distributivo finitamente multivalente. Deixe-se E representar uma qualquer (a) $FND_Q(S)$, (b) $fnd_{Q,u}(S)$, (c) $FNC_Q(S)$, ou (d) $fnc_{Q,u}(S)$, em que S é um sinal e u é um valor de verdade. Então, para todo Q, todo S e todo u a regra $S[(Qx)\,A(x)] \Longrightarrow E$ é correta e redutora. O número de disjuntos e conjuntos em (a) e (c) ((b) e (d)) é limitado por $2^{|W|}$ ($2^{|W|-1}$, respetivamente).

4.2.4.4 Regras de transformação e preservação de estrutura

As regras de transformação acima podem corromper a estrutura das fórmulas originais, tanto nas lógicas multivalentes como em LC. Com efeito, estas regras de

transformação preservam somente a linguagem no sentido em que no processo de transformação não se adicionam novos símbolos às fórmulas originais. Alternativamente, podemos aplicar regras de transformação preservadoras de estrutura, as quais codificam a estrutura das fórmulas originais pela introdução de símbolos de predicados. Por esta razão podemos chamar-lhes *regras de introdução de predicados*.

4.2.32. DEFINIÇÃO. As regras de introdução de predicados têm a forma

$$\text{(IP)} \qquad S\left[\phi\left(A\left(\vec{x}\right)\right)\right] \Longrightarrow S\left[\phi\left(P\left(\vec{x}\right)\right)\right] \wedge \bigwedge_{v_i \in W} \left(\forall \vec{x}\right)\left(\left\{v_i\right\}\left[A\left(\vec{x}\right)\right] \equiv \left\{v_i\right\}\left[P\left(\vec{x}\right)\right]\right)$$

$$\text{(IP}') \qquad S\left[\phi\left(A\left(\vec{x}\right)\right)\right] \Longrightarrow S\left[\phi\left(P\left(\vec{x}\right)\right)\right] \wedge \bigwedge_{v_i \in W} \left(\forall \vec{x}\right)\left(\left\{v_i\right\}\left[A\left(\vec{x}\right)\right] \rightarrow \left\{v_i\right\}\left[P\left(\vec{x}\right)\right]\right)$$

em que ϕ é uma fórmula multivalente, A é uma subfórmula contida em ϕ numa ou mais posições e $\vec{x}$ é o vetor das variáveis livres em A. Quanto a $\phi\left(P\left(\vec{x}\right)\right)$, em que P é um novo símbolo de predicado que não ocorre em nenhum outro lugar em ϕ, denota a substituição por $P\left(\vec{x}\right)$ em ϕ de uma ou mais ocorrências de A.

Na regra (IP), $\left(\left\{v_i\right\}\left[A\left(\vec{x}\right)\right] \equiv \left\{v_i\right\}\left[P\left(\vec{x}\right)\right]\right)$ garante que em todas as interpretações que satisfazem esta equivalência se dá o caso que, para todo x, $P\left(\vec{x}\right)$ e $A\left(\vec{x}\right)$ assumem o mesmo valor de verdade $v_i \in W$. Note-se que se obtém (IP$'$) precisamente pela remoção de um dos dois conjuntos logicamente equivalentes

$$\bigwedge_{v_i \in W} \left(\forall \vec{x}\right)\left(\left\{v_i\right\}\left[A\left(\vec{x}\right)\right] \rightarrow \left\{v_i\right\}\left[P\left(\vec{x}\right)\right]\right) \wedge \bigwedge_{v_i \in W} \left(\forall \vec{x}\right)\left(\left\{v_i\right\}\left[P\left(\vec{x}\right)\right] \rightarrow \left\{v_i\right\}\left[A\left(\vec{x}\right)\right]\right).$$

É evidente que a introdução de novos símbolos de predicados implica que as regras (IP) e (IP$'$) não são estritamente corretas, mas ambos os lados de (IP) e (IP$'$) são equisatisfazíveis, nomeadamente para fins de resolução. Contudo, estas regras produzem um número bastante inferior de cláusulas em comparação com as regras preservadoras de linguagem. De facto, uma fórmula com m ocorrências de conetores com uma aridade máxima n e l ocorrências de quantificadores contém não mais do que $m|W|^n + l2^{|W|} + 1$ cláusulas. De novo se verifica a necessidade de encontrar um equilíbrio entre propriedades das lógicas multivalentes.

4.2.4.5 Tradução para forma clausal

Tal como em LC, os procedimentos de demonstração por resolução aplicados à verificação da validade de uma fórmula multivalorada ϕ requerem que ϕ seja traduzida

para forma clausal. Visto que a resolução é um método de refutação, o primeiro passo no procedimento da demonstração é a transformação de $\neg S\left[\phi\right] = \overline{S}\left[\phi\right]$, em que S é tipicamente o conjunto de valores designados (i.e., ϕ é verdadeira se toma um dos valores de verdade em S), num conjunto de cláusulas por (1) eliminação dos conetores e quantificadores multivalentes, (2) eliminação de todos os quantificadores existenciais pela substituição de quaisquer variáveis quantificadas existencialmente por termos de Skolem e (3) obtenção de FNCs pela aplicação repetida da lei da distributividade.

Porém, antes destas operações é necessário computar as regras de transformação da lógica em consideração. Estas computações são relativamente fáceis dadas as especificações dos conetores e quantificadores, bem como de outras propriedades do sistema em consideração (ex.: ordem em W; conjunto D de valores designados), podendo ser computadas por meios automáticos (cf. o sistema MULTlog; Baaz et al., 1996). (Obviamente, a computação das regras de transformação para W com elevada cardinalidade é tudo menos trivial para i específico, $0 < i < 1$.)

4.2.33. Exemplo. As seguintes são as regras de transformação para Ł_n (assinala-se v_i com asterisco sempre que $v_i \neq \text{V}, \text{F}$):

$$
\begin{array}{lcl}
\{v_i\}\left[A \wedge B\right] & \Longrightarrow & \{\{v_j\}\left[A\right], \{v_j\}\left[B\right], \{\{v_i\}\left[A\right], \{v_i\}\left[B\right]\}\}, j \geq i \\
\{v_i\}\left[A \vee B\right] & \Longrightarrow & \{\{v_j\}\left[A\right], \{v_j\}\left[B\right], \{\{v_i\}\left[A\right], \{v_i\}\left[B\right]\}\}, j \leq i \\
\{\text{V}\}\left[A \to B\right] & \Longrightarrow & \{\{v_j\}\left[A\right], \{v_k\}\left[B\right]\}, j < c \leq k, 1 \leq c \leq n-1 \\
\{\text{F}\}\left[A \to B\right] & \Longrightarrow & \{\{\text{V}\}\left[A\right], \{\text{F}\}\left[B\right]\} \\
\{v_i^*\}\left[A \to B\right] & \Longrightarrow & \{\{v_k\}\left[A\right], \{v_s\}\left[B\right]\}, k \neq r \text{ e } (n-1) - (r-s) \neq i, \\
 & & 0 \leq r \leq n-1 \\
\{\text{V}\}\left[(\forall x)\, A\left(x\right)\right] & \Longrightarrow & \{(\forall x)\, \{\text{V}\}\left[A\left(x\right)\right]\} \\
\{\text{F}\}\left[(\forall x)\, A\left(x\right)\right] & \Longrightarrow & \{(\exists x)\, \{\text{F}\}\left[A\left(x\right)\right]\} \\
\{v_i^*\}\left[(\forall x)\, A\left(x\right)\right] & \Longrightarrow & \{(\forall x)\, \{v_j\}\left[A\left(x\right)\right], (\exists x)\, \{v_i\}\left[A\left(x\right)\right]\}, j \geq i \\
\{\text{V}\}\left[(\exists x)\, A\left(x\right)\right] & \Longrightarrow & \{(\exists x)\, \{\text{V}\}\left[A\left(x\right)\right]\} \\
\{\text{F}\}\left[(\exists x)\, A\left(x\right)\right] & \Longrightarrow & \{(\forall x)\, \{\text{F}\}\left[A\left(x\right)\right]\} \\
\{v_i^*\}\left[(\exists x)\, A\left(x\right)\right] & \Longrightarrow & \{(\forall x)\, \{v_j\}\left[A\left(x\right)\right], (\exists x)\, \{v_i\}\left[A\left(x\right)\right]\}, j \leq i \\
\{v_i\}\left[\neg A\right] & \Longrightarrow & \left\{\left\{v_{(n-1)-i}\right\}\left[A\right]\right\}
\end{array}
$$

4.2.34. Exemplo. Considere-se a fórmula $A = (\forall x)\, P\left(x\right) \to (\exists y)\, P\left(y\right)$ em qŁ_3 com $W = \{\text{V}, \text{I}, \text{F}\}$ como $\{\text{I}\}\left[A\right]$. No exemplo 4.2.27 obtivemos o resultado

$$\{\text{I}\}\left[A \to B\right] \equiv (\{\text{V}\}\left[A\right] \vee \{\text{F}\}\left[B\right]) \wedge (\{\text{I}\}\left[A\right] \vee \{\text{I}\}\left[B\right]).$$

Logo,

$$\{\mathrm{I}\}\,[A] \equiv (\{\mathrm{V}\}\,[(\forall x)\,P\,(x)] \vee \{\mathrm{F}\}\,[(\exists y)\,P\,(y)]) \wedge (\{\mathrm{I}\}\,[(\forall x)\,P\,(x)] \vee \{\mathrm{I}\}\,[(\exists y)\,P\,(y)])\,.$$

Mostramos o processo de transformação em forma clausal, indicando as principais operações por meio de números (ver abaixo) e sublinhando as partes específicas sobre as quais essas operações incidem no caso de só parte da expressão sofrer essa operação.

$$\{\mathrm{I}\}\,[A] \equiv (\{\mathrm{V}\}\,[(\forall x)\,P\,(x)] \vee \{\mathrm{F}\}\,[(\exists y)\,P\,(y)]) \wedge (\{\mathrm{I}\}\,[(\forall x)\,P\,(x)] \vee \{\mathrm{I}\}\,[(\exists y)\,P\,(y)])$$

$$\equiv \Big(\underline{(\forall x)\,\{\mathrm{V}\}\,[P\,(x)]} \vee \{\mathrm{F}\}\,[(\exists y)\,P\,(y)]\Big) \wedge (\{\mathrm{I}\}\,[(\forall x)\,P\,(x)] \vee \{\mathrm{I}\}\,[(\exists y)\,P\,(y)]) \tag{1}$$

$$\equiv ((\forall x)\,\{\mathrm{V}\}\,[P\,(x)] \vee \{\mathrm{F}\}\,[(\exists y)\,P\,(y)]) \wedge \Big(\underline{((\exists x)\,\{\mathrm{I}\}\,P\,(x) \wedge (\forall x)\,\{\mathrm{V},\mathrm{I}\}\,[P\,(x)])} \vee \{\mathrm{I}\}\,[(\exists y)\,P\,(y)]\Big) \tag{2}$$

$$\equiv ((\forall x)\,\{\mathrm{V}\}\,[P\,(x)] \vee \{\mathrm{F}\}\,[(\exists y)\,P\,(y)]) \wedge \underline{((\exists x)\,\{\mathrm{I}\}\,[P\,(x)] \vee \{\mathrm{I}\}\,[(\exists y)\,P\,(y)]) \wedge} \\ \underline{([(\forall x)\,\{\mathrm{V},\mathrm{I}\}\,[P\,(x)]] \vee \{\mathrm{I}\}\,[(\exists y)\,P\,(y)])} \tag{3}$$

$$\equiv \Big((\forall x)\,\{\mathrm{V}\}\,[P\,(x)] \vee \underline{(\forall y)\,\{\mathrm{F}\}\,[P\,(y)]}\Big) \wedge \tag{4}$$
$$\Big((\exists x)\,\{\mathrm{I}\}\,[P\,(x)] \vee \underline{((\exists y)\,\{\mathrm{I}\}\,[P\,(y)] \wedge (\forall y)\,\{\mathrm{I},\mathrm{F}\}\,[P\,(y)])}\Big) \wedge \tag{5}$$
$$\Big(((\forall x)\,\{\mathrm{V},\mathrm{I}\}\,[P\,(x)]) \vee \underline{((\exists y)\,\{\mathrm{I}\}\,[P\,(y)] \wedge (\forall y)\,\{\mathrm{I},\mathrm{F}\}\,[P\,(y)])}\Big) \tag{5}$$

$$\equiv ((\forall x)\,\{\mathrm{V}\}\,[P\,(x)] \vee (\forall y)\,\{\mathrm{F}\}\,[P\,(y)]) \wedge \\ \underline{((\exists x)\,\{\mathrm{I}\}\,[P\,(x)]) \wedge ((\exists y)\,\{\mathrm{I}\}\,[P\,(y)] \vee ((\forall y)\,\{\mathrm{I},\mathrm{F}\}\,[P\,(y)]))\wedge} \\ \underline{(((\forall x)\,\{\mathrm{V},\mathrm{I}\}\,[P\,(x)]) \vee (\exists y)\,\{\mathrm{I}\}\,[P\,(y)]) \wedge} \\ \underline{(((\forall x)\,\{\mathrm{V},\mathrm{I}\}\,[P\,(x)]) \vee ((\forall y)\,\{\mathrm{I},\mathrm{F}\}\,[P\,(y)]))} \tag{3),(6}$$

$$\equiv ((\forall x)\,\{\mathrm{V}\}\,[P\,(x)] \vee (\forall y)\,\{\mathrm{F}\}\,[P\,(y)]) \wedge \\ (\exists x)\,\{\mathrm{I}\}\,[P\,(x)] \wedge (((\forall x)\,\{\mathrm{V},\mathrm{I}\}\,[P\,(x)]) \vee ((\forall y)\,\{\mathrm{I},\mathrm{F}\}\,[P\,(y)])) \tag{7}$$

$$\equiv_{sat} (\{\mathrm{V}\}\,[P\,(x)] \vee \{\mathrm{F}\}\,[P\,(y)]) \wedge \underline{(\{\mathrm{I}\}\,[P\,(a)])} \wedge \\ (\{\mathrm{V},\mathrm{I}\}\,[P\,(x)] \vee \{\mathrm{I},\mathrm{F}\}\,[P\,(y)]) \tag{8}$$

As operações indicadas são as seguintes:

(1) $\{\mathrm{V}\}\,[(\forall x)\,A\,(x)] \Longrightarrow (\forall x)\,\{\mathrm{V}\}\,[A\,(x)]$
(2) $\{\mathrm{I}\}\,[(\forall x)\,A\,(x)] \Longrightarrow (\exists x)\,\{\mathrm{I}\}\,[A\,(x)] \wedge (\forall x)\,\{\mathrm{V},\mathrm{I}\}\,[A\,(x)]$
(3) Lei da distributividade
(4) $\{\mathrm{F}\}\,[(\exists x)\,A\,(x)] \Longrightarrow (\forall x)\,\{\mathrm{F}\}\,[A\,(x)]$
(5) $\{\mathrm{I}\}\,[(\exists x)\,A\,(x)] \Longrightarrow (\exists x)\,\{\mathrm{I}\}\,[A\,(x)] \wedge (\forall x)\,\{\mathrm{I},\mathrm{F}\}\,[A\,(x)]$
(6) Contração de disjunções de EFAs idênticas a menos de renomeação de variáveis
(7) Remoção do $3^{\underline{o}}$ e do $4^{\underline{o}}$ conjuntos, redundantes em presença do $2^{\underline{o}}$ conjunto
(8) Skolemização

4.3 Resolução assinalada

4.3.1 Regras de inferência

Há duas regras que constituem um cálculo de resolução direto e completo em termos de refutação para cláusulas assinaladas e/ou fórmulas FNCs assinaladas:[3]

4.3.1. Definição. Seja $S_i\,[P]$ um literal assinalado e denote-se por C ou C_i uma qualquer cláusula. Então, (R1) e (R2) são duas regras de inferência do cálculo de resolução assinalada:

$$\text{(R1)} \qquad \frac{S_1\,[P] \vee C_1 \quad S_2\,[P] \vee C_2}{(S_1 \cap S_2)\,[P] \vee C_1 \vee C_2}$$

$$\text{(R2)} \qquad \frac{\{\}\,[P] \vee C}{C}$$

(R1) é a *resolução binária assinalada* e $(S_1 \cap S_2)\,[P] \vee C_1 \vee C_2$ é uma *resolvente binária*. (R2) é a regra de *simplificação*: diz-se que uma cláusula C é uma simplificação de C' se C resulta de C' pela eliminação de todos os literais com o sinal vazio. Note-se que em (R1), se $\{\} \in (S_1 \cap S_2)\,[P]$, então $(S_1 \cap S_2)\,[P]$ é insatisfazível e pode ser removido da resolvente.

4.3.2. Exemplo. Seja $W = \{0, 1, 2\}$. Aplicamos (R1) nos exemplos seguintes:

[3]Para uma demonstração: Ansótegui et al. (2002).

(1)
$$\frac{\{2\}\,[P] \quad \overline{\{2\}\,[P]}}{\square}$$

(2)
$$\frac{\{0,1\}\,[P] \vee \{2\}\,[Q] \quad \overline{\{1\}\,[P]}}{\{0\}\,[P] \vee \{2\}\,[Q]}$$

Lembra-se que uma resolvente $\{\}\,[P]$ é equivalente a $\square$.

As técnicas de resolução assinalada datam basicamente do início dos anos 1990 (ex.: Murray & Rosenthal, 1993; Hähnle, 1994), mas na verdade pode-se dizer que os primeiros passos em direção à lógica assinalada (e consequentemente à resolução assinalada) foram dados por Rescher (1969) com o conceito de conjunto de valores de verdade antidesignados. Como se viu (Secção 2.2.), pode-se selecionar conjuntos de valores designados e antidesignados de um qualquer sistema lógico finitamente multivalente de tal modo que se trabalhe com o menor número de conjuntos possíveis – nomeadamente, com apenas dois conjuntos, D^+ e D^-, os quais se pode obviamente associar à verdade e à falsidade, respetivamente. Isto permite a aplicação direta das técnicas de resolução de LC às lógicas finitamente multivalentes. Com efeito, dadas duas cláusulas $D^+\,[P] \vee A$ e $D^-\,[P] \vee B$, obtém-se, pela aplicação de (R1), a resolvente $A \vee B$ se $D^+ \cap D^- = \emptyset$.

Contudo, como Caleiro & Marcos (2009) fazem notar, para sistemas lógicos *genuinamente n-valentes* (sistemas para os quais n é a cardinalidade da menor coleção de valores de verdade v_n graças à qual eles têm uma semântica verofuncional) esta caracterização bivalente implica a perda da caracterização verofuncional. Se se optar por um compromisso entre expressividade e complexidade, então é-se obrigado a abordagens que salvaguardam em maior ou menor grau a multivalência. Como se disse, os sistemas assinalados diferem em pontos importantes dos sistemas multivalentes cujas fórmulas se testa em termos de validade "dentro" daqueles, mas eles são equivalentes em termos de satisfazibilidade. Embora do ponto de vista da dedução automática a maior vantagem da lógica assinalada seja a de permitir a aplicação das técnicas de resolução de LC (a qual é aliás o caso mais simples de um cálculo de resolução) às lógicas multivalentes (as quais, como se viu, generalizam LC), tal não acarreta uma redução completa destas últimas à LC.

Com respeito à aplicação de (R1), é óbvio que, e contrariamente à resolução clássica, o literal resolvido pode permanecer (como um *resíduo*). Tal evita-se facilmente pela aplicação da regra (R3).

4.3.3. DEFINIÇÃO. A regra de inferência (R3), definida pela seguinte formulação

$$\text{(R3)} \qquad \frac{S_1\,[P] \vee C_1 ... S_m\,[P] \vee C_m}{C_1 \vee ... \vee C_m} \quad \text{se} \bigcap_{1 \leq i \leq m} S_i = \emptyset$$

chama-se *resolução paralela assinalada*.

4.3.4. EXEMPLO. Seja $W = \{a, b, c, d\}$. Aplicamos (R3) no seguinte exemplo:

$$\frac{\{a\}\,[P] \vee \{a\}\,[Q] \quad \{b, c\}\,[P] \vee \{c\}\,[R] \quad \{d\}\,[P]}{\{a\}\,[Q] \vee \{c\}\,[R]}$$

Dado um conjunto de valores totalmente ordenado, a completude da resolução binária assinalada é preservada para as fórmulas FNCMs e FNCRs se se simplicar (R1) como

$$\text{(R4)} \qquad \frac{S_1\,[P] \vee C_1 \quad S_2\,[P] \vee C_2}{C_1 \vee C_2} \quad \text{se } S_1 \cap S_2 = \emptyset.$$

4.3.5. DEFINIÇÃO. (R4) chama-se *resolução binária monoassinalada/regular*.

4.3.6. EXEMPLO. Seja $W = \left\{0, \frac{1}{2}, 1\right\}$. Então a seguinte é uma aplicação de (R4):

$$\frac{\{0\}\,[P] \vee \left\{\frac{1}{2}\right\}[Q] \quad \{1\}\,[P] \vee \{1\}\,[R]}{\left\{\frac{1}{2}\right\}[Q] \vee \{1\}\,[R]}$$

Obviamente, (R4) evita a geração do resíduo permitida por (R1).

Embora (R5) não seja necessária para fins de completude, é útil para a redução do espaço de procura (bem como para simplificar a demonstração da completude). Basicamente, (R5) une literais com o mesmo átomo:

$$\text{(R5)} \qquad \frac{S_1\,[P] \vee ... \vee S_m\,[P] \vee C}{(S_1 \cup ... \cup S_m)\,[P] \vee C}.$$

4.3.7. DEFINIÇÃO. (R5) é a regra de *fusão* (*merging*); C chama-se uma versão em *forma normal* ou *normalizada* de C' se resulta de C' pela aplicação repetida (i.e., tantas vezes quanto possível) de (R2) e (R5) a C'. (O resultado desta estratégia é a remoção de todos os literais com sinais vazios; além disso, os átomos são diferentes em todos os diferentes literais em C.)

A extensão das regras acima aos sistemas assinalados de primeira ordem é imediata, bastando para tal a introdução das técnicas de unificação, i.e., de um umg σ, pelo que as seguintes regras de inferência são válidas no cálculo de resolução assinalada:

$$\text{(R1}') \qquad \frac{S_1\,[P_1] \vee C_1 \quad S_2\,[P_2] \vee C_2}{((S_1 \cap S_2)\,[P_1] \vee C_1 \vee C_2)\,\sigma}, \quad \sigma = umg\,(P_1, P_2)$$

$$\text{(R3}') \qquad \frac{S_1\,[P_1] \vee C_1 ... S_m\,[P_m] \vee C_m}{(C_1 \vee ... \vee C_m)\,\sigma} \quad \text{se} \bigcap_{1 \leq i \leq m} S_i = \emptyset,\ \sigma = umg\,(P_1, ..., P_m)$$

$$\text{(R4}') \qquad \frac{S_1\,[P_1] \vee C_1 \quad S_2\,[P_2] \vee C_2}{(C_1 \vee C_2)\,\sigma} \quad \text{se } S_1 \cap S_2 = \emptyset,\ \sigma = umg\,(P_1, P_2)$$

$$\text{(R5}') \qquad \frac{S_1\,[P_1] \vee ... \vee S_m\,[P_m] \vee C}{((S_1 \cup ... \cup S_m)\,[P_1] \vee C)\,\sigma}, \quad \sigma = umg\,(P_1, .., P_m)$$

A validade das regras de inferência acima deve-se obviamente à generalização do teorema de Herbrand (teorema 3.2.6) a conjuntos de cláusulas assinaladas.

4.3.8. Proposição. Um conjunto $\mathcal{C}$ de cláusulas assinaladas é insatisfazível sse é H-insatisfazível.

Demonstração. ($\Rightarrow$) Pela definição 3.2.11 e pela definição de conjunto de valores designados (cf. em especial def. 1.3.5), é óbvio que um qualquer conjunto de cláusulas assinaladas $\mathcal{C}$ é H-insatisfazível se não existe nenhuma interpretação H tal que $val_{H\mathcal{I}}\,(C_i) \in D$ para toda a cláusula assinalada C_i. Logo, e ainda pela definição 1.2.5, se $\mathcal{C}$ é H-insatisfazível, então é insatisfazível.

($\Leftarrow$) Para um qualquer conjunto de cláusulas $\mathcal{C}$ toda a interpretação $\mathcal{I}$ induz uma interpretação $H\mathcal{I} = \{\{v\}\,[P] \mid val_{\mathcal{I}}\,(P) = v, P \in H\,(\mathcal{C})\}$. Obviamente, se $\mathcal{I}$ não satisfaz $\mathcal{C}$, então $H\mathcal{I}$ não satisfaz $\mathcal{C}$. □

4.3.9. Exemplo. Sejam $\mathcal{C} = \{\{\{\mathrm{V}\}[Q(a)]\}, \{\{\mathrm{F}\}[Q(x)], \{\mathrm{I}\}[Q(x)], \{\mathrm{V}\}[Q(f(x))]\}\}$, $W = \{\mathrm{V}, \mathrm{I}, \mathrm{F}\}$ e $D = \{\mathrm{V}\}$. Então,

$H_{\mathcal{C}} = \{a, f(a), f(f(a)), ...\}$
$H(\mathcal{C}) = \{Q(a), Q(f(a)), Q(f(f(a))), ...\}$.

A interpretação H $\{\{\mathrm{V}\}[Q(a)], \{\mathrm{V}\}[Q(f(a))], \{\mathrm{V}\}[Q(f(f(a)))], ...\}$ é satisfazível; pelo contrário, a interpretação H $\{\{\mathrm{V}\}[Q(a)], \{\mathrm{F}\}[Q(f(a))], \{\mathrm{V}\}[Q(f(f(a)))], ...\}$ é insatisfazível.

É possível combinar as regras de inferência acima; por exemplo, a regra seguinte, uma combinação de uma série de passos de (R1) e (R2), é um refinamento de resolução assinalada.

4.3.10. Definição. A regra de inferência

$$\text{(R6)} \qquad \frac{S_1[P_1] \vee C_1 ... S_k[P_k] \vee C_k}{(C_1 \cup ... \cup C_k)\sigma}, \quad \bigcap_{1 \leq i \leq k} S_i = \emptyset,\ \sigma = umg(P_i(1 \leq i \leq k))$$

chama-se *macro-resolução* (ou *hiper-resolução*) e $(C_1 \cup ... \cup C_k)\sigma$ chama-se a *macro-resolvente* (a *hiper-resolvente*, respetivamente).

4.3.11. Exemplo. (Ver exemplo 4.3.16)

A dupla terminologia da definição 4.3.10 deve-se ao facto de a hiper-resolução ser uma versão da macro-resolução, a qual visa a redução do número de resolventes pela resolução simultânea de várias cláusulas (cf. Secção 3.4.2).

A importância da aplicabilidade das interpretações H às lógicas multivalentes vê-se ainda no facto que a base de Herbrand permite um meio para um outro refinamento da resolução assinalada, tal como o faz no caso de LC. Falamos aqui da resolução por ordenamento de átomos (*A-ordering*): porque se refere a átomos (vs. literais), pode ser aplicada diretamente a muitas cláusulas.

4.3.12. Definição. A regra de inferência

$$\text{(R7)} \qquad \frac{S_1[P_1] \vee C_1 ... S_k[P_k] \vee C_k}{(C_1 \cup ... \cup C_k)\sigma}, \quad \bigcap_{1 \leq i \leq k} S_i = \emptyset,\ \sigma = umg(P_i(1 \leq i \leq k)),$$

$$P_i\sigma \not<_A Q \text{ para todo } R\,[Q] \in C_i\sigma$$

chama-se *macro-resolução por ordenamento* $<_A$, e a conclusão chama-se *macro-resolvente por ordenamento* $<_A$.

Tal como no caso de LC (cf. Secção 3.4.1), é necessário especificar o ordenamento a que são sujeitos os átomos, sendo que este incide tipicamente sobre a profundidade (ϑ) dos termos e das variáveis dos átomos, estipulando-se que o átomo resolvido não é menor do que um qualquer literal na macro-resolvente.

4.3.2 Resolução assinalada para lógicas infinitamente multivalentes

As regras de inferência acima são imediatamente aplicáveis às lógicas *finitamente* multivalentes. No caso das lógicas *infinitamente* multivalentes, impõem-se algumas considerações. Falamos aqui sobretudo das lógicas difusas abordadas acima nas secções 2.4.4 e 2.5.2. Estas lógicas têm aplicações fundamentais em múltiplas áreas; como tal, a sua axiomatização e automatização em termos de dedução mostram-se essenciais. Porém, como se viu acima, existem problemas por resolver no que respeita à axiomatização destas lógicas, o que se reflete diretamente na campo da automatização, pouco desenvolvido até agora. Isto vale também para a resolução assinalada. Por exemplo, Baaz et al. (2001), um trabalho que de certo modo cristaliza ainda o estado da arte em resolução assinalada para lógicas multivalentes, mais não faz do que apresentar algumas curtas notas sobre este tópico, nenhuma delas apontando para uma aplicação deste cálculo. Apresentamos aqui algumas sugestões para a aplicação da resolução assinalada a estas lógicas.

LC pode ser considerada como o caso não trivial mais simples de LD (obviamente, um sistema lógico com $|W| = 1$ é trivial). Com efeito, os conetores de LD são normais (cf. def. 2.3.1 e prop. 2.3.2), pelo que a restrição a um conjunto de valores de verdade $W = \{0, 1\}$ por $1 \in D^+$ e $0 \in D^-$ resulta num sistema clássico (cf. ainda resultados 2.4.42-5). Sabemos já que a lógica assinalada permite o raciocínio clássico em lógicas multivalentes, pelo que uma aplicação da resolução assinalada a LD aparece em princípio como possível. Vejamos como.

Podemos, por exemplo, com base nas definições 2.4.29 e 2.4.30 começar por determinar que $\varepsilon \in [0.5, 1]$. Isto permite-nos uma redução direta a LC por $\varepsilon\,[\phi] \equiv \mathrm{V}\,[\phi]$ e logo $\neg\varepsilon\,[\phi] \equiv \mathrm{F}\,[\phi]$ em que $\neg\varepsilon \in [0, 0.5)$ e ϕ é ou não quantificada. Uma tal redução permitir-nos-ia uma implementação direta da resolução assinalada num demonstrador automático como o Prover9, uma vez que $\mathrm{V}\,[\phi] \equiv \phi$ e $\mathrm{F}\,[\phi] \equiv \neg\phi$. Uma tal

implementação seria a do exemplo 3.5.2, uma vez que, a ser rigorosos, o predicado *is much smaller than* faz do problema posto por Schubert um *teorema difuso* (cf. exemplo 2.5.16). Com efeito, no exemplo 3.5.2 denotámos esta propriedade pelo predicado binário $Smaller\,(x,y)$, mas mais correto teria sido a denotação por $Ms\,(x,y)$ do exemplo 2.5.16, uma vez que *much* é um modificador difuso da propriedade *is smaller than*. Entre a valoração $val_{\mathcal{I}}\,(Ms\,(x,y)) = 1$ e $val_{\mathcal{I}}\,(Ms\,(x,y)) = 0$, i.e., entre a interpretação que atribui a x a propriedade de ser muito mais pequeno do que y sem restrições (ou seja, absolutamente) e a interpretação que recusa absolutamente a x a propriedade de ser muito mais pequeno do que y (ou seja, x não é decididamente muito mais pequeno do y), existe uma infinidade de graus de *vagueza*.

Contudo, uma redução $W_{LD} = \{0,1\}$ ou mesmo uma redução de $|W_{LD}| = n$ para um qualquer n finito implica a perda da capacidade de um sistema lógico de lidar com a vagueza, a motivação fundamental nas bases matemáticas (os conjuntos difusos) da criação das lógicas difusas (cf. Zadeh, 1965; 1975). Além disso, a tradução de fórmulas de LD para lógica clausal é de modo geral inexequível, apenas com alguns fragmentos a permitir uma tal tradução. Por exemplo, Mundici & Olivetti (1998) demonstraram que a resolução e a construção de modelos para pelo menos a lógica infinitamente multivalente de Łukasiewicz são tratáveis polinomialmente nos fragmentos dados por cláusulas de Horn e cláusulas de Krom (cláusulas com no máximo dois literais). Baseiam-se para tal numa noção de *base* ε (traduzindo θ-*support*) de um literal não negativo L como sendo o conjunto $base_{\varepsilon}\,(L) = \{val^* \mid val\,(L) \geq \varepsilon\}$ em que $\varepsilon \in (0,1]$ e um conjunto de cláusulas $\mathcal{C}$ é ε-*satisfazível* sse existe uma qualquer atribuição $val^* : Vp \longrightarrow [0,1]$. A noção de uma *complementaridade* ε para literais L e M é evidente a partir disto, e Mundici e Olivetti formulam então uma regra de resolução binária para $Ł_{\aleph}$ para duas cláusulas $C_1 = L_1, ..., L_n$ e $C_2 = M_1, ..., M_q$ e uma variável proposicional p tais que (i) p ocorre apenas no literal positivo L_i de C_1, (ii) p ocorre apenas no literal negativo M_j de C_2, (iii) L_i e M_j são ε-complementares. Satisfeitas estas condições, C_1 e C_2 são então p-*resolúveis com respeito a* ε na implementação da regra

$$\frac{C_1 \quad C_2}{(C_1 - \{L_1\}) \cup (C_2 - \{M_j\})}.$$

É óbvio que a satisfazibilidade ε permite uma caracterização de p como um literal assinalado regular, i.e. como $\uparrow i\,[p]$ ou $\downarrow i\,[p]$, mas a tradução para lógica clausal baseia-se em & (em vez de $\wedge$) e em funções de McNaughton de uma variável, representando um procedimento de demonstração interno (cf. Baaz et al., 2001). Porém, a tradução para lógica clausal de fórmulas de $Ł_{\aleph}$ por este procedimento está longe de ser trivial (cf. Metcalfe et al., 2009, pp. 159ss).

Uma **função de McNaughton de n variáveis** é uma função $f : [0,1]^n \to [0,1]$ que satisfaz as seguintes condições:
(1) f é contínua;
(2) existem polinómios lineares $p_1, ..., p_k$ tais que para cada $1 \leq i \leq k$, $b_i, m_i \in \mathbb{Z}$ e para cada $x \in [0,1]^n$ existe um índice j com $f(x) = p_j(n)$.

Beckert et al. (1999) apresentam um método baseado em W com uma ordem parcial $\geq$, i.e. um reticulado $(W, \geq)$, a partir do qual se obtêm procedimentos de resolução para fórmulas assinaladas regulares em FNC. O método vale apenas para cláusulas de Horn assinaladas e é interessante para nós na medida em que permite a aplicação dos resultados obtidos a reticulados de valores de verdade infinitos.

4.3.13. Definição. Seja $\mathbb{A}$ uma fórmula de Horn regular sobre um reticulado (possivelmente infinito) $(W, \geq)$ com ínfimo $\bot$ e supremo $\top$; definimos $(W_{\mathbb{A}}, \geq)$ como sendo o sub-reticulado de $(W, \geq)$ gerado pelos elementos $\{i \in W \mid i \text{ ocorre em } \mathbb{A}\} \cup \{\bot\}$.

4.3.14. Teorema. $\mathbb{A}$ é satisfazível por uma interpretação para $(W, \geq)$ sse é satisfazível no reticulado $(W_{\mathbb{A}}, \geq)$.

A demonstração ($\Rightarrow$) é trivial (toda a interpretação para $(W_{\mathbb{A}}, \geq)$ é uma interpretação para $(W, \geq)$ e para($\Leftarrow$) basta demonstrar que para todos os valores de verdade em $\mathbb{A}$ se verifica que

$$\mathcal{I} \models \uparrow i\,[p] \quad \text{sse} \quad \mathcal{I}_{\mathbb{A}} \models \uparrow i\,[p]$$

o que se faz com base num operador *supremum* $\sqcup$ comum aos reticulados $(W, \geq)$ e $(W_{\mathbb{A}}, \geq)$. Tem-se então as regras de inferência

$$(\mathfrak{R}1) \quad \frac{\begin{array}{c} \to\uparrow i\,[p] \\ \uparrow i_1\,[p_1], ..., \uparrow i_l\,[p_l], ..., \uparrow i_k\,[p_k] \to\uparrow j\,[q] \end{array}}{\uparrow i_1\,[p_1], ..., \uparrow i_{l-1}\,[p_{l-1}], \uparrow i_{l+1}\,[p_{l+1}] ..., \uparrow i_k\,[p_k] \to\uparrow j\,[q]}$$

e

$$(\mathfrak{R}2) \quad \frac{\begin{array}{c} \to i\,[p] \\ \to j\,[q] \end{array}}{\to\uparrow (i \sqcup j)\,[p]}.$$

($\mathfrak{R}1$) chama-se *resolução unitária regular positiva* e requer a condição que $i \geq i_l$; quanto a ($\mathfrak{R}2$), chama-se *redução regular* e requer que nem $i \geq j$ nem $j \geq i$. Este procedimento parece aplicável em princípio às lógicas difusas (ou a fragmentos de algumas delas) ao fazer-se coincidir $\bot$ ($\top$) com 0 (1, respetivamente).

Faz-se notar que as sugestões acima valem *apenas* para as versões proposicionais das lógicas de interesse. Até ao presente, não se concebeu nenhum procedimento de demonstração eficiente para as lógicas difusas de primeira ordem.

4.3.3 Correção, completude e teorema principal da resolução assinalada

Uma vez que a vantagem da lógica clausal assinalada é a de permitir o raciocínio clássico em lógicas multivalentes em termos do cálculo de resolução, convém que a generalização de dedução por resolução em lógicas multivalentes seja o mais conservadora possível. Como se pode ver facilmente por uma comparação da definição seguinte com a definição 3.3.4, a noção fundamental de *dedução por resolução* em LC é imediatamente generalizável às lógicas multivalentes:

4.3.15. DEFINIÇÃO. Dados um conjunto de cláusulas $\mathcal{C}$ e uma cláusula C, dizemos que C é *derivável* de $\mathcal{C}$ por *resolução multivalente*, denotado por $\mathcal{C} \vdash_{resmv} C$, se existir um número finito de cláusulas $C_1, ..., C_n = C$ tal que para cada C_i, $1 \leq i \leq n$,

1. C_i é uma variante de uma cláusula em $\mathcal{C}$ ou
2. C_i é uma resolvente de variantes de cláusulas C_j e C_k, $j, k < i$.

A nossa abordagem à resolução assinalada atribui um papel importante à seleção de um conjunto de valores designados, nomeadamente porque a situa teoricamente no contexto do SAT e do problema da decisão. Porém, verifica-se que nas aplicações práticas da resolução assinalada o interesse recai frequentemente em demonstrar ou refutar asserções do género "$val_{\mathcal{I}}(\phi) \in U$ para toda $\mathcal{I}$" dado um qualquer conjunto $U \subset W$. Assim sendo, voltamos por agora a nossa atenção para este aspeto; veremos em seguida qual o impacto deste aspeto na demonstração da *(quase-)validade* de fórmulas multivaloradas como "$val_{\mathcal{I}}(\phi) \in D$ para toda $\mathcal{I}$" dado $D \subset W$.

4.3.16. EXEMPLO. Seja dadas a fórmula $A = (\forall x) P(x) \rightarrow (\exists y) P(y)$ e a fórmula assinalada $\{\mathrm{I}\}[A] = \{\mathrm{I}\}[(\forall x) P(x) \rightarrow (\exists y) P(y)]$ em qŁ$_3$. A fórmula A é fechada e queremos saber se A pode de facto tomar o valor de verdade I em qŁ$_3$. Começamos

por obter as cláusulas de $\{\mathrm{I}\}[A]$ (cf. exemplo 4.2.34) $C_1 = \{\mathrm{V}\}[P(x)] \vee \{\mathrm{F}\}[P(y)]$, $C_2 = \{\mathrm{I}\}[P(a)]$ e $C_3 = \{\mathrm{V},\mathrm{F}\}[P(x)] \vee \{\mathrm{I},\mathrm{F}\}[P(y)]$ e temos então o conjunto $\mathcal{C}_{\{\mathrm{I}\}[A]} = \{C_1, C_2, C_3\}$. Note-se que se mostrarmos pela aplicação das regras (R1)-(R7) que o conjunto de cláusulas $\mathcal{C}_{\{\mathrm{I}\}[A]}$ é insatisfazível, i.e., $\mathcal{C}_{\{\mathrm{I}\}[A]} \vdash_{resmv} \square$, então mostramos que a fórmula A não pode tomar o valor de verdade I em qŁ$_3$, ou seja, $\{\mathrm{I}\}[A]$ não é uma fórmula válida em qŁ$_3$.

C_1	$\{\mathrm{V}\}[P(x)] \vee \{\mathrm{F}\}[P(y)]$	
C_2	$\{\mathrm{I}\}[P(a)]$	
C_3	$\{\mathrm{V},\mathrm{F}\}[P(x)] \vee \{\mathrm{I},\mathrm{F}\}[P(y)]$	
C_4	$\{\}[P(a)] \vee \{\mathrm{F}\}[P(y)]$	Resolvente de $C_1\theta$ e $C_2\theta$, $\theta = \{x \mapsto a\}$
C_5	$\{\mathrm{V}\}[P(x)] \vee \{\}[P(a)]$	Resolvente de $C_1\lambda$ e $C_2\lambda$, $\lambda = \{y \mapsto a\}$
C_6	$\{\mathrm{F}\}[P(y_1)]$	C_4, (R2) e renomeação
C_7	$\{\mathrm{V}\}[P(x_1)]$	C_5, (R2) e renomeação
C_8	$\square$	Resolvente de $C_6\sigma$ e $C_7\sigma$, $\sigma = \{x_1 \mapsto c, y_1 \mapsto c\}$, por (R3)

$\{\mathrm{I}\}[A]$ foi refutada; logo, $(\forall x)\,P(x) \to (\exists y)\,P(y)$ não pode tomar o valor de verdade I em qŁ$_3$. Note-se que C_3 foi removida da resolução acima pela definição 4.2.14, pontos (i) e (iii), uma vez efetuada a condensação de C_3. Note-se ainda que podemos obter diretamente C_8 de $C_1\sigma$ e $C_2\sigma$ por hiper-resolução, $\sigma = \{x \mapsto a, y \mapsto a\}$, σ é um umg de C_1 e C_2.

A noção de árvore semântica (def. 3.2.14) mostrou-se útil para demonstrar a completude da resolução em LC. Ora, dá-se o caso que também esta noção é imediatamente generalizável às lógicas multivalentes, nomeadamente por meio da lógica clausal assinalada. Lembramos que dado um qualquer conjunto de cláusulas $\mathcal{C}$, numa árvore semântica $T_{\mathcal{C}}$ cada ramo $I(N)$ é uma *atribuição de valores de verdade* aos átomos básicos de $H(\mathcal{C})$. Seja $A(\mathcal{C})$ o conjunto de átomos básicos das cláusulas básicas de $\mathcal{C}$, i.e., $A(\mathcal{C}) \subseteq H(\mathcal{C})$: então, cada ramo $I(N)$ em $T_{\mathcal{C}}$ contém pela menos uma atribuição de valores de verdade a $A(\mathcal{C})$ e cada atribuição de valores de verdade completa a $A(\mathcal{C})$ (i.e., uma interpretação H) é um subconjunto da união do conjunto de literais nas arestas de $I(N)$. Recorde-se ainda, do ponto (2) da definição 3.2.14, que N é um nó de insucesso se $I(N)$ falsifica uma qualquer instância básica de uma qualquer cláusula C de $\mathcal{C}$, mas $I(N')$ não falsifica nenhuma instância básica

de alguma cláusula C de $\mathcal{C}$. Temos então que uma árvore semântica de $\mathcal{C}$ é fechada se para todo $I(N)$ se verifica que a intersecção das FNCs obtidas de $\mathcal{C}$ com as FNDs correspondentes que constituem as arestas de $T_{\mathcal{C}}$ (cf. prop. 3.2.14(1-i)) é vazia, i.e., se para cada $I(N)$ e todo o átomo A_i de uma cláusula qualquer C se verifica que

$$\left(\bigvee_{i=1}^{n} A_i\right) \cap \left(\bigwedge_{i=1}^{n} \neg A_i\right) = \emptyset$$

Efetivamente, $\bigwedge_{i=1}^{n} \neg A_i$ é uma refutação de $\bigvee_{i=1}^{n} A_i$ (cf. prop. 1.1.27). Seja agora S um sinal de um átomo $A_i \in A(\mathcal{C})$; é óbvio que para cada $I(N)$ de uma árvore semântica de $\mathcal{C}$ se verifica que

$$\left(\bigvee_{i=1}^{n} S[A_i]\right) \cap \left(\bigwedge_{i=1}^{n} \overline{S}[A_i]\right) = \emptyset$$

Logo, $I(N)$ omite pelo menos uma atribuição de um valor de verdade $v_i \in S$ a um conjunto de literais de uma qualquer cláusula C e de igual modo ao conjunto de átomos da cláusula básica correspondente a C, pelo que cada $I(N)$ omite exatamente uma interpretação H de $\mathcal{C}$.

O teorema seguinte precisa esta generalização no que respeita à insatisfazibilidade H.

4.3.17. Teorema. Um conjunto de cláusulas $\mathcal{C}$ é H-insatisfazível sse existe um subconjunto finito $A(\mathcal{C}) \subseteq H(\mathcal{C})$ tal que toda e qualquer árvore semântica de $A(\mathcal{C})$ é fechada para $\mathcal{C}$.

Demonstração. ($\Rightarrow$) Seja $\mathcal{C}$ um conjunto de cláusulas insatisfazível. Pelo teorema 3.2.17, existe uma árvore semântica finita fechada de $\mathcal{C}$. Seja $T_{\mathcal{C}}$ uma árvore semântica fechada de $\mathcal{C}$. Sendo fechada, todos os ramos de $T_{\mathcal{C}}$ são fechados, i.e., terminam num nó de insucesso (cf. def. 3.2.14(2)). Pela definição de árvore semântica, o número de nós em $T_{\mathcal{C}}$ que se encontram acima de um nó de insucesso de uma árvore semântica finita é igualmente finito. Isto implica que para cada conjunto finito de cláusulas insatisfazível $\mathcal{C}$ existe um conjunto finito $\mathcal{C}'$ de cláusulas básicas construídas a partir de $H(\mathcal{C})$ que é H-insatisfazível. Logo, toda a árvore semântica fechada $T'_{\mathcal{C}}$ de $\mathcal{C}'$ implica que $T_{\mathcal{C}}$ é de igual modo fechada, pelo que para construir uma árvore fechada de $\mathcal{C}$ basta um subconjunto finito $A(\mathcal{C}) \subseteq H(\mathcal{C})$, $A(\mathcal{C})$ é o conjunto de átomos básicos em $\mathcal{C}'$.

($\Leftarrow$) Seja $T_{\mathcal{C}}$ uma árvore semântica fechada para um subconjunto finito $A(\mathcal{C}) \subseteq$

$H(\mathcal{C})$. Seja $\mathcal{C}'$ o conjunto de cláusulas básicas de $\mathcal{C}$ e seja $A(\mathcal{C})$ o conjunto de átomos básicos de $\mathcal{C}'$. Suponha-se agora que $\mathcal{C}'$ é H-satisfazível. Então, existe uma árvore semântica aberta $T'_{\mathcal{C}}$ de $\mathcal{C}'$. Isto implica que há pelo menos uma instância básica em $\mathcal{C}'$ de $\mathcal{C}$ que não é falsificada por uma interpretação H num nó de insucesso de $T'_{\mathcal{C}}$. Logo, existe pelo menos um ramo de $T_{\mathcal{C}}$ que não termina num nó de insucesso. Mas isto vai contra a assunção que $T_{\mathcal{C}}$, uma árvore semântica de $A(\mathcal{C}) \subseteq H(\mathcal{C})$, é fechada. Logo, $\mathcal{C}$ deve ser H-insatisfazível. □

4.3.18. Teorema (correção do cálculo de resolução para lógicas multivalentes). Para um qualquer conjunto de cláusulas $\mathcal{C}$, se $\mathcal{C} \vdash_{resmv} \square$, então $\mathcal{C}$ é H-insatisfazível.

Demonstração. Viu-se já no caso de LC que não há nenhuma interpretação que satisfaça a cláusula vazia. Logo, $\mathcal{C}$ é insatisfazível sempre que $\square$ é derivável. Além disso, uma vez que $\square$ não tem nenhum átomo básico pertencente a $A(\mathcal{C}) \subseteq H(\mathcal{C})$ que possa ser satisfeito por uma interpretação H, temos que se $\square$ é derivável de $\mathcal{C}$, então $\mathcal{C}$ é H-insatisfazível, nomeadamente através de um subconjunto $\mathcal{C}' \subseteq \mathcal{C}$, $\mathcal{C}'$ é o conjunto de cláusulas básicas de $\mathcal{C}$. □

4.3.19. Teorema (completude do cálculo de resolução para lógicas multivalentes). Para um qualquer conjunto de cláusulas $\mathcal{C}$, se $\mathcal{C}$ é H-insatisfazível, então $\mathcal{C} \vdash_{resmv} \square$.

Demonstração. Sabemos já que se $\mathcal{C}$ é um conjunto insatisfazível de cláusulas, então todas as árvores semânticas de $A(\mathcal{C})$ são fechadas. Nomeadamente, toda a árvore semântica $T'_{\mathcal{C}}$ é fechada para $\mathcal{C}$. Seja $\mathcal{C}'$ o conjunto de cláusulas básicas assinaladas de $\mathcal{C}$. Selecionamos uma árvore $T'_{\mathcal{C}}$ cujas etiquetas são literais monoassinalados. Começamos pelo nó de insucesso N de $T'_{\mathcal{C}}$; visto que é um nó de insucesso, temos que, dadas duas cláusulas $C_1 \in \mathcal{C}$ e $C_2 \in \mathcal{C}$ e uma substituição básica σ, $(C_1 - L_1)\sigma \cup (C_2 - L_2)\sigma$ é um subconjunto do conjunto de refutação de N para $L_1\sigma = \{v\}[A]$ e $L_2\sigma = \{u\}[A]$, $v \neq u$, $v, u \in W$. Pela definição de árvore semântica fechada, existe uma resolvente C_3 de (fatores de) C_1 e C_2 que falham em N, e um nó de insucesso para $\mathcal{C} \cup \{C_3\}$ só pode encontrar-se em pelo menos $I(N) - 2N'$ acima de N. Para C_n, n é o número de resolventes em $\mathcal{C}$, é então óbvio que o nó de insucesso para $\mathcal{C} \cup \{C_n\}$ se situa na raiz de $T'_{\mathcal{C}}$, correspondente à cláusula vazia. □

4.3.20. Teorema (teorema principal da resolução assinalada para lógicas multivalentes). Seja ϕ uma fórmula fechada e seja $\mathcal{C}_{\overline{U}\Phi}$ o conjunto de cláusulas da tradução

clausal $\overline{U}\Phi$ de $v\,[\phi]$ para um qualquer valor de verdade $v \in \overline{U}$, $U \subset W$. Então, todas as interpretações atribuem um valor de verdade $u \in U$ a ϕ sse $\mathcal{C}_{\overline{U}\Phi} \vdash_{resmv} \square$, em que *resmv* designa uma qualquer das regras (R1)-(R7).

Demonstração. ($\Rightarrow$) Dada uma fórmula ϕ e um conjunto de valores de verdade $U \subset W$, se todas as interpretações atribuem um valor de verdade $u \in U$ a ϕ então $v\,[\phi]$ para um qualquer valor de verdade $v \in \overline{U}$ é insatisfazível pela definição 1.2.5 e pelas proposições 4.2.26 e 4.2.31 a tradução clausal $\overline{U}\Phi$ de $v\,[\phi]$ é insatisfazível. Seja que $\mathcal{C}_{\overline{U}\Phi}$ é o conjunto de cláusulas correspondente a $\overline{U}\Phi$; então, pela proposição 4.3.8 $\mathcal{C}_{\overline{U}\Phi}$ é H-insatisfazível. Logo, pelos teoremas 4.3.19 temos que $\mathcal{C}_{\overline{U}\Phi} \vdash_{resmv} \square$.

($\Leftarrow$) Pelo teorema 4.3.18, se $\mathcal{C}_{\overline{U}\Phi} \vdash_{resmv} \square$, então $\mathcal{C}_{\overline{U}\Phi}$ é H-insatisfazível. Pela proposição 4.3.8 então $\mathcal{C}_{\overline{U}\Phi}$ é insatisfazível. Logo, pelo teorema 1.2.8, temos que $\models u\,[\phi]$, i.e., todas as interpretações atribuem um valor de verdade $u \in U$ a ϕ. $\square$

Temos então o seguinte resultado fundamental:

4.3.21. PROPOSIÇÃO. O algoritmo seguinte constitui uma automatização da demonstração de teoremas pelo cálculo de resolução em lógicas multivalentes.

Dada uma qualquer fórmula ϕ num sistema lógico multivalente S com um conjunto de valores de verdade W:

1. Formar a forma clausal $\overline{D}\Phi$ da fórmula assinalada $v\,[\phi]$, $v \in \overline{D}$, em que $D \subset W$ e D é o conjunto de valores designados.
2. Obter o conjunto de cláusulas $\mathcal{C}_{\overline{D}\Phi}$ a partir de $\overline{D}\Phi$.
3. Aplicar o cálculo de resolução multivalente (regras (R1)-(R7)) a $\mathcal{C}_{\overline{D}\Phi}$ de modo a verificar a insatisfazibilidade: se $\mathcal{C}_{\overline{D}\Phi}$ é insatisfazível, então $u\,[\phi]$, $u \in D$, é uma fórmula válida em S.

4.3.22. EXEMPLO. Seja dada a fórmula $\{\mathrm{I}\}\,[(\forall x)\,P\,(x) \rightarrow (\exists y)\,P\,(y)]$ do exemplo 4.3.16 e seja que $D_{\mathrm{qŁ_3}} = \{\mathrm{V}\}$. Vimos que a fórmula $A = (\forall x)\,P\,(x) \rightarrow (\exists y)\,P\,(y)$ não pode tomar o valor de verdade I em qŁ$_3$. Obtemos então uma demonstração de $\models_{\mathrm{qŁ_3}} \{\mathrm{V}\}\,[A]$, ou seja, da validade de $\{\mathrm{V}\}\,[A]$ em qŁ$_3$.

4.3.23. TEOREMA. Dada uma fórmula ϕ numa qualquer lógica multivalente em que $D \subset W$ e D é o conjunto de valores designados, a tradução de uma qualquer

fórmula assinalada $v\,[\phi]$, $v \in \overline{D}$, para a EFA $\overline{D}\Phi$ e para o respetivo conjunto de cláusulas $\mathcal{C}_{\overline{D}\Phi}$ combinada com a respetiva refutação é uma demonstração da validade de $u\,[\phi]$, $u \in D$.

Demonstração. A demonstração é trivial dados os resultados acima. □

Concluímos que a resolução assinalada é um cálculo adequado (i.e., correto e completo) para a demonstração automática de teoremas em lógicas finitamente multivalentes.

4.4 Exercícios

1. Compute as FNCs das seguintes fórmulas nos sistemas B_3^I, B_3^E, K_3^S, K_3^W e P_3:

   (a) $\{\mathrm{V}\}\,[A \to B]$
   (b) $\{\mathrm{I}\}\,[A \to B]$
   (c) $\{\mathrm{F}\}\,[A \to B]$
   (d) $\{\mathrm{I}\}\,[A \wedge B]$
   (e) $\{\mathrm{V}\}\,[A \vee B]$
   (f) $\{\mathrm{F}\}\,[(A \wedge B) \to C]$

2. Compute as FNCs das seguintes fórmulas em qŁ$_3$.

   (a) $\{\mathrm{I}\}\,[(\forall y)\,P\,(y) \to (\forall x)\,Q\,(x)]$
   (b) $\{\mathrm{V}\}[(\exists x)\,R\,(x) \wedge \neg\,(\forall x)\,R\,(x)]$
   (c) $\{\mathrm{F}\}[(\forall x)\,\neg P\,(x) \vee (\exists y)\,R\,(y)]$

3. Determine a satisfazibilidade das fórmulas obtidas no exercício 2. Que conclui em relação à validade das fórmulas originais?

4. Como se poderá representar uma árvore semântica completa fechada para um conjunto de cláusulas numa lógica trivalente?

Bibliografia

1. Ansótegui, C., Béjar, R., Cabiscol, A., Li, C. M., & Manyà, F. (2002). Resolution methods for many-valued CNF formulas. In *Fifth International Symposium on the Theory and Applications of Satisfiability Testing, SAT-2002*, Cincinnati, EUA (pp. 152-163).

2. Aristóteles (DI/1941). *De Interpretatione*. Tr. E. M. Edgehill, in R. McKeon (org.), *The Basic Works of Aristotle*. Nova Iorque: Random House.

3. Baader, F. & Snyder, W. (2001). Unification theory. In A. Robinson & A. Voronkov (orgs.), *Handbook of automated reasoning*, vol. 1 (p. 441-526). Amsterdão: Elsevier / Cambridge, MA: MIT Press.

4. Baaz, M., Fermüller, C. G., & Salzer, G. (2001). Automated deduction for many-valued logics. In A. Robinson & A. Voronkov (orgs.), *Handbook of automated reasoning*, vol. II (p. 1357-1400). Amsterdão: Elsevier / Cambridge, MA: MIT Press.

5. Baaz, M., Fermüller, C. G., Salzer, G., & Zach, R. (1996). MULtlog 1.0: Towards an expert system for many-valued logics. In *13th International Conference on Automated Deduction (CADE'96)*, *Lecture Notes in Artificial Intelligence, 1104*, 226-230.

6. Beckert, B., Hähnle, R., & Manyà, F. (1999). Transformations between signed and classical clause logic. In *Proceedings of the 29th International Symposium on Multiple-Valued Logic* (pp. 248-255). Los Alamitos: IEEE CS Press.

7. Beckert, B., Hähnle, R., & Manyà, F. (2000). The SAT problem of signed CNF formulas. In D. Basin, M. D'Agostino, D. M. Gabbay, S. Matthews, & L. Viganò (orgs.), *Labelled deduction* (pp. 59-80). Dordrecht: Kluwer.

8. Bellman, R. E. & Zadeh, L. A. (1977). Local and fuzzy logics. In J. M. Dunn and G. Epstein (orgs.), *Modern uses of multiple-valued logic* (p. 105-165). Dordrecht: Reidel.

9. Belnap, N. (1977a). How a computer should think. In G. Ryle (ed.), *Contemporary Aspects of Philosophy* (pp. 30-55). Stocksfield: Oriel Press.

10. Belnap, N. (1977b). A useful four-valued logic. In J. M. Dunn & G. Epstein (orgs.), *Modern uses of multiple-valued logic* (p. 8-37). Dordrecht: Reidel.

11. Bergmann, M. (2008). *An introduction to many-valued and fuzzy logic. Semantics, algebras, and derivation systems.* Cambridge: Cambridge University Press.

12. Beth, E. W. (1955). Semantic entailment and formal derivability. *Mededlingen der Koninklijke Nederlandse Akademie van Wetenschappen*, *18*, 309-342.

13. Blake, A. (1937). *Canonical expressions in Boolean algebra.* Tese de doutoramento. University of Chicago, Illinois.

14. Bochvar, D. A. (1939). Ob odnom tréhznačnom isčislénii i égo priménénii k analizu paradosov klassičéskogo rasširennogo funkcjonal'nogo isčislénia. *Matématičéskij Sbornik*, 4, 287-308. (Tr. ingl.: On a three-valued calculus and its application to the analysis of paradoxes of classical extended functional calculus. Tr. de M. Bergmann. *History and Philosophy of Logic*, *2* (1981), 87-112.)

15. Bolc, L. & Borowik, P. (1992). *Many-valued logics 1: Theoretical foundations.* Berlim, etc.: Springer.

16. Bolc, L. & Borowik, P. (2003). *Many-valued logics 2: Automated reasoning and practical applications.* Berlim, etc.: Springer.

17. Caleiro, C. & Marcos, J. (2009). Classic-like analytic tableaux for finite-valued logics. In H. Ono, M. Kanazawa, R. de Queiroz (orgs.), *Logic, Language, Information, and Computation. Proceedings of the 16th International Workshop, WoLLIC*, Tóquio, Japão (pp. 268-280). Springer.

18. Caleiro, C., Carnielli, W., Coniglio, M. E., & Marcos, J. (2007). Two's company: "The humbug of many logical values". In J.-Y. Béziau (org.), *Logica universalis* (pp. 175-194). 2ª ed. Basileia: Birkhäuser.

19. Chang, C.-L. & Lee, R. C.-T. (1973). *Symbolic logic and mechanical theorem proving.* Nova Iorque e Londres: Academic Press.

20. Church, A. (1936a). An unsolvable problem of elementary number theory. *American Journal of Mathematics*, *58*, 345-363.

21. Church, A. (1936b). A note on the Entscheidungsproblem. *Journal of Symbolic Logic*, *1*, 40-41.

22. Cook, S. A. (1971). The complexity of theorem proving procedures. *Proceedings of the 3rd Annual ACM Symposium of Theory of Computing*, 151-158.

23. Davis, M. (1957). A computer program for Presburger's algorithm. In *Summaries of Talks Presented at the Summer Institute for Symbolic Logic* (pp. 215-233). Princeton, NJ: Institute for Defense Analysis.

24. Davis, M. & Putnam, H. (1958). Computational methods in the propositional calculus. Rensselaer Polytechnic Institute. (Relatório não publicado.)

25. Davis, M. & Putnam, H. (1960). A computing procedure for quantification theory. *Journal of the ACM*, *7*, 201-215.

26. Davis, M., Logemann, G., & Loveland, D. (1962). A machine program for theorem-proving. *Communications of the ACM*, *5*, 394-397.

27. Di Zenzo, S. (1988). A many-valued logic for approximate reasoning. *IBM Journal of Research and Development*, *32*, 552-565.

28. Epstein, G. (1993). *Multiple-valued logic design: An introduction*. Bristol & Filadélfia: Institute of Physics Publishing.

29. Esteva, F., Godo, L., Hájek, P., & Montagna, F. (2003). Hoops and fuzzy logic. *Journal of Logic and Computation*, *13*, 532-555.

30. Fitting, M. & Orłowska, E. (orgs.) (2003). *Beyond two: Theory and applications of multiple-valued logic*. Heidelberg: Physica-Verlag.

31. Fortnow, L. (2009). The status of the P versus NP problem. *Communications of the ACM*, *52*, 78-86.

32. Frege, G. (1892). Über Sinn und Bedeutung. *Zeitschrift für Philosophie und philosophische Kritik C*, 25-50.

33. Gilmore, P. (1960). A proof method for quantification theory: Its justification and realization. *IBM Journal of Research and Development*, *4*, 28-35.

34. Gottwald, S. (2001). *A treatise on many-valued logics*. Filadélfia: Research Studies Press.

35. Grigolia, G. (1977). Algebraic analysis of Łukasiewicz-Tarski's n-valued logical systems. In R. Wójcicki & G. Malinowski (orgs.), *Selected papers on Łukasiewicz sentential calculi* (p. 81-92). Wrocław: Ossolineum.

36. Hähnle, R. (1994). *Automated deduction in multiple-valued logics*. Oxford: Oxford University Press.

37. Hájek, P. (1997). Fuzzy logic and arithmetical hierarchy II. *Studia Logica*, *58*, 129-141.

38. Hájek, P. (1998). *Metamathematics of fuzzy logic*. Dordrecht: Kluwer.

39. Harrison, J. (2009). *Handbook of practical logic and automated reasoning*. Cambridge, etc.: Cambridge University Press.

40. Herbrand, J. (1930). *Recherches sur la théorie de la démonstration*. Thèses présentées à la Faculté des Sciences de Paris. 128 p.

41. Hilbert, D. (1900). Mathematische Probleme. Vortrag, gehalten auf dem internationalen Mathematiker-Kongress zu Paris 1900. *Nachrichten von der Königl. Gesellschaft der Wissenschaften zu Göttingen. Mathematisch-Physikalische Klasse*, 253-297. (Tr. ingl.: Mathematical problems. *Bulletin of the American Mathematical Society*, *8* (1902), 437-479.)

42. Hilbert, D. & Ackermann, W. (1928). *Grundzüge der theoretischen Logik*. Berlim: Springer.

43. Hilbert, D. & Bernays, P. (1934). *Grundlagen der Mathematik*. Vol. I. Berlim: Springer.

44. Hilbert, D. & Bernays, P. (1939). *Grundlagen der Mathematik*. Vol. II. Berlim: Springer.

45. Hintikka, K. J. J. (1955). Form and content in quantification theory. *Acta Philosophica Fennica*, *8*, 7-55.

46. Hoder, K. & Voronkov, A. (2009). Comparing unification algorithms in first-order theorem proving. In *KI'09 Proceedings of the 22nd Annual German Conference on Advances in Artificial Intelligence* (pp. 435-443). Berlim, Heidelberg: Springer.

47. Kalman, J. A. (2001). *Automated reasoning with OTTER*. Princeton, NJ: Rinton Press.

48. Karpenko, A. S. (2006). *Łukasiewicz logics and prime numbers*. Beckington: Luniver Press.

49. Kleene, S. (1938). On a notation for ordinal numbers. *Journal of Symbolic Logic*, *3*, 150-155.

50. Kleene, S. C. (1952). *Introduction to metamathematics*. Amsterdão: North.Holland.

51. Lee, R. C. T. (1972). Fuzzy logic and the resolution principle. *Journal of the ACM*, *19*, 109-119.

52. Leitsch, A. (1997). *The resolution calculus*. Berlim, etc.: Springer.

53. Łukasiewicz, J. (1920). O logice trójwartościowej. *Ruch Filozoficzny*, *5*, 170-171. (Tr. ingl.: On three-valued logic. In L. Borkowski (org.), *Selected Works* (p. 87-88). Amsterdão: North-Holland.)

54. Łukasiewicz, J. (1930). Philosophische Bemerkungen zu mehrwertigen Systemen des Aussagenkalküls. *Comptes Rendus des Scéances de la Société des Sciences et des Lettres de Varsovie CL. III*, *23*, 51-77. (Tr. ingl.: Philosophical remarks on many-valued systems of propositional logic. In S. McCall (org.), *Polish Logic 1920-1939* (p. 40-65). Oxford: Clarendon Press. 1967.)

55. Łukasiewicz, J., & Tarski, A. (1930). Untersuchungen über den Aussagenkalkül. [Tr. tít.: Investigações sobre o cálculo proposicional]. *Comptes Rendus des Scéances de la Société des Sciences et des Lettres de Varsovie CL. III*, *23*, 30-50.

56. Malinowski, G. (1989). *Equivalence in intensional logics*. Varsóvia: Polish Academy of Sciences, Institute of Philosophy and Sociology.

57. Malinowski, G. (1993). *Many-valued logics*. Oxford: Clarendon Press.

58. Malinowski, G. (2008). Fregean axiom and many-valuedness. *Bulletin of the Section of Logic*, *37*, 245-252.

59. Malinowski, G. (2012). Multiplying logical values. *Logical Investigations*, 18, 292-308.

60. Mendelson, E. (2009). *Introduction to mathematical logic*. 5ª ed. Chapman and Hall / CRC.

61. Metcalfe, G., Olivetti, N., & Gabbay, D. (2009). *Proof theory for fuzzy logics*. Springer.

62. Morgan, C. G. (1976). A resolution principle for a class of many-valued logics. *Logique et Analyse*, *19*, 311-339.

63. Mostowski, A. (1957). On a generalization of quantifiers. *Fundamenta Mathematicae*, *44*, 12-36.

64. Mundici, D. & Olivetti, N. (1998). Resolution and model building in the infinite-valued calculus of Łukasiewicz. *Theoretical Computer Science*, *200*, 335-366.

65. Murray, N. V. & Rosenthal, E. (1993). Signed formulas: A liftable meta-logic for multiple-valued logics. In *Proceedings of the International Symposium on Methodologies for Intelligent Systems (ISMIS)*, Trondheim, Noruega (pp. 275-284). Springer.

66. Newell, A., Shaw, J., & Simon, H. (1957). Empirical explorations with the Logic Theory Machine. *Proceedings of the West Joint Computer Conference*, *15*, 218-239.

67. Novák, V. (1990). On the syntactic-semantical completeness of first-order fuzzy logic. (Partes I e II.) *Kybernetika*, *26*, 47-66; 134-154.

68. Novák, V., Perfilieva, I., & Močkoř, J. (1999). *Mathematical principles of fuzzy logic*. Boston: Kluwer.

69. Orłowska, E. (1978). The resolution principle for ω^+-valued logic. *Fundamenta Informaticae*, *II*, 1-15.

70. Pavelka, J. M. (1979). On fuzzy logic (Partes I, II, III). *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, *25*, 45-52; 119-134; 447-464.

71. Post, E. L. (1921). Introduction to a general theory of elementary propositions. *American Journal of Mathematics*, *43*, 163-185.

72. Prawitz, D. (1960). An improved proof procedure. *Theoria*, *26*, 102-139.

73. Priest, G. (2008). *An introduction to non-classical logic*. Cambridge: Cambridge University Press.

74. Quine, W. V. O. (1955). A proof procedure for quantification theory. *Journal of Symbolic Logic*, *20*, 141-149.

75. Rasiowa, H. (1974). *An algebraic approach to non-classical logics*. Amsterdão: North-Holland.

76. Rescher, N. (1969). *Many-valued logic*. McGraw-Hill.

77. Robinson, A. J. (1965). A machine-oriented logic based on the resolution principle. *Journal of the ACM*, *12*, 23-41.

78. Rosser, J. B. & Turquette, A. R. (1952). *Many-valued logics*. Amsterdam: North-Holland.

79. Scarpelini, B. (1962). Die Nichtaxiomatisierbarkeit des unendlichwertigen Prädikatenkalküls von Łukasiewicz. [Tr. tít.: A não axiomatizabilidade do cálculo de predicados infinitamente valente de Łukasiewicz.] *Journal of Symbolic Logic*, *17*, 159-170.

80. Schmitt, P. H. (1986). Computational aspects of three-valued logic. In J. H. Siekmann (org.), *Proceedings of the 8th International Conference on Automated Deduction*, Oxford, Inglaterra (pp. 190-198). Berlim, Heidelberg: Springer.

81. Schütte, K. (1956). Ein System des verknüpfenden Schliessens. *Archiv für mathematische Logik und Grundlagen der Wissenschaft*, *2*, 55-67.

82. Skolem, T. (1920). Logisch-kombinatorische Untersuchungen über die Erfüllbarkeit oder Beweisbarkeit mathematischer Sätze. [Tr. tít.: Investigações lógico-combinatórias acerca da completude ou demonstrabilidade de proposições matemáticas.] *Skrifter utgit av Videnskapsselskapet i Kristiania*, *4*, 4-36.

83. Słupecki, J. (1936). Der volle dreiwertige Aussagenkalkül. *Comptes Rendus des Scéances de la Société des Sciences et des Lettres de Varsovie Cl. III*, *29*, 9-11. (Tr. ingl.: The full three-valued propositional calculus. In S. McCall (org.), *Polish Logic 1920-1939* (p. 335-337). Oxford: Clarendon Press. 1967.)

84. Słupecki, J. (1939a). Um critério para a completude de sistemas lógicos proposicionais multivalentes [Tr. tít.]. *Comptes Rendus des Séances de la Société des Sciences et des Lettres de Varsovie Cl. III*, *32*, 102-109.

85. Słupecki, J. (1939b). Demonstração da axiomatizabilidade de sistemas de cálculo proposicional multivalentes completos [Tr. tít.]. *Comptes Rendus des Séances de la Société des Sciences et des Lettres de Varsovie Cl. III*, *32*, 110-128.

86. Sofronie-Stokkermans, V. (1998). On translation of finitely-valued logics to classical first-order logic. In H. Prade (org.), *Proceedings of the 13th European*

Conference on Artificial Intelligence (ECAI), Brighton, Reino Unido (pp. 410-411). John Wiley & Sons.

87. Stachniak, Z. (1996). *Resolution proof systems: An algebraic approach.* Kluwer.

88. Suszko, R. (1957). Uma teoria formal de valores lógicos [Tr. tít.]. *Studia Logica*, *6*, 145-320.

89. Suszko, R. (1977). The Fregean axiom and Polish mathematical logic in the 1920's. *Studia Logica*, *36*, 373-380.

90. Tseitin, G. S. (1968). On the complexity of derivations in the propositional calculus. In A. O. Slisenko (org.), *Studies in constructive mathematics and mathematical logic. Part II. Seminar in mathematics* (pp. 115-125) [tr. do russo para inglês]. Steklov Mathematical Institute.

91. Turing, A. (1936-7). On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society, Series 2*, *41*, 230-265.

92. Wajsberg, M. (1931). Aksjomatyzacja trójwartościowego rachunku zdań.*Comptes Rendus des Scéances de la Société des Sciences et des Lettres de Varsovie Cl. III*, *24*, 126-148. (Tr. ingl.: Axiomatization of the three-valued propositional calculus. In S. McCall (org.), *Polish Logic 1920-1939* (p. 264-284). Oxford: Clarendon Press. 1967.)

93. Walther, C. (1985). A mechanical solution of Schubert's Steamroller by many-sorted resolution. *Artificial Intelligence*, *26*, 217-224.

94. Whitehead, A. N. & Russell, B. (1910). *Principia mathematica.* Vol. I. Cambridge: Cambridge University Press.

95. Wójcicki, R. (1969). Logical matrices strongly adequate for structural sentential calculi. *Bulletin de l'Académie Polonaise des Sciences, Série des sci. math. astr. et physiques*, *6*, 333-335.

96. Zach, R. (1993). *Proof theory of finite-valued logics.* Monografia / Relatório técnico TUW-E185.2-Z.1-93, Institut für Algebra und diskrete Mathematik, TU Viena.

97. Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, *8*, 338-353.

98. Zadeh, L. A. (1975). Fuzzy logic and approximate reasoning. *Synthese*, *30*, 407-428.

Índice remissivo

Índice de nomes

Indica-se apenas nomes ligados a aspetos fundamentais abordados no texto (ex.: história, sistemas lógicos, teoremas, etc.). Para outros autores citados, ver Bibliografia.

Índice de conteúdos

As lógicas multivalentes fazem parte do vasto campo das lógicas não clássicas. Apresentando como característica principal o facto de possuírem conjuntos de valores de verdade com mais do que os dois valores clássicos (verdadeiro e falso), têm uma importância central em campos como as ciências dos computadores e a inteligência artificial, convidando ainda a problemáticas filosóficas relacionadas com o conceito de valores de verdade.

Nesta obra aborda-se as lógicas multivalentes mais relevantes de um ponto de vista matemático e computacional, adotando uma perspetiva dos fundamentos a par com aplicações práticas. No que respeita a primeira, a noção algébrica de matriz lógica é aqui central. Tendo a questão prática hoje fulcral da automatização do raciocínio em consideração, oferece-se ao leitor um cálculo para a automatização da demonstração de teoremas em lógicas multivalentes — a resolução assinalada — que pode ser facilmente implementado por um computador humano ou num demonstrador automático.

Trata-se de um texto autossuficiente em que são revistas ou tratadas em detalhe todas as temáticas essenciais à compreensão dos fundamentos matemáticos e computacionais das lógicas multivalentes e se dá ao leitor a possibilidade de realizar inúmeros exercícios práticos de revisão e consolidação.

Sobre o autor:
http://luismaugusto.my-free.website
http://luisaugustophil.50webs.com