Revisão de crenças em lógicas de descrição e em outras lógicas não clássicas

Márcio Moretto Ribeiro

Tese apresentada
ao
Instituto de Matemática e Estatística
da
Universidade de São Paulo
para
obtenção do título
de
Doutor em Ciências

Programa: Ciência da Computação
Orientadora: Profa. Dra. Renata Wassermann

Durante o desenvolvimento deste trabalho o autor recebeu auxílio financeiro da FAPESP

São Paulo, agosto de 2010

Revisão de crenças
em lógicas de descrição
e em outras lógicas não clássicas

Esta versão definitiva da tese
contém as correções e alterações sugeridas pela
Comissão Julgadora durante a defesa realizada
por (Márcio Moretto Ribeiro) em 20/09/2010.

Comissão Julgadora:

- Profa. Dra. Renata Wassermann (orientadora) - IME-USP
- Prof. Dr. Marcelo Finger - IME-USP
- Prof. Dr. Fabio Gagliardi Cozman - EP-USP
- Prof. Dr. Walter Alexandre Carnielli - IFCH/CLE-UNICAMP
- Prof. Dr. Mario Roberto Folhadela Benevides - COPPE-UFRJ

Resumo

A área de revisão de crenças estuda como agentes racionais mudam suas crenças ao receberem novas informações. O marco da área de revisão de crenças foi a publicação do trabalho de Alchourrón, Gärdenfors e Makinson [AGM85]. Nesse trabalho conhecido como paradigma AGM foram definidos critérios de racionalidade para tipos de mudança de crenças. Desde então, a área de revisão de crenças foi influenciada por diversas disciplinas como filosofia, computação e direito.

Paralelamente ao desenvolvimento da área de revisão de crenças, os últimos 20 anos foram marcados por um grande avanço no estudo das lógicas de descrição. Tal avanço, impulsionado pelo desenvolvimento da web-semântica, levou a adoção de linguagens inspiradas em lógicas de descrição (OWL) como padrão para se representar ontologias na web.

Nessa tese tratamos do problema de aplicar a teoria da revisão de crenças a lógicas não clássicas e especialmente a lógicas de descrição. Trabalhos recentes mostraram que o paradigma AGM é incompatível com diversas lógicas de descrição. Estendemos esses resultados mostrando outras lógicas que não são compatíveis com o paradigma AGM. Propomos formas de aplicar a teoria de revisão tanto em bases quanto em conjuntos de crenças a essas lógicas. Além disso, usamos algoritmos conhecidos da área de depuração de ontologias para implementar operações em bases de crenças.

Palavras-chave: revisão de crenças, representação de conhecimento, inteligência artificial, lógicas não clássicas, lógicas de descrição, ontologias.

Abstract

Belief revision theory studies how rational agents change their beliefs after receiving new information. The most influential work in this area is the paper of Alchourrón, Gärdenfors and Makinson [AGM85]. In this work, known as AGM paradigm rationality criteria for belief change were defined. Since then, the field has been influenced by many areas like philosophy, computer science and law.

Parallel to the development of belief revision field, in the past 20 years there was a huge grow in the study of description logics. The climax of this development was the adoption of OWL (a language based on description logics) as the standard language to represent ontologies on the web.

In this work we deal with the problem of applying belief revision in to non-classical logics, specially description logics. Recent works showed that the AGM paradigm is not compliant with several description logics. We have extended this work by showing that other logics are not compliant with AGM paradigm. Furthermore, we propose alternative ways to apply belief revision techniques to these logics. Finally, we show that well known algorithms from the area of ontology debugging field can be used to implement the proposed constructions.

Keywords: belief revision, knownledge representation, artificial inteligence, non-classical logics, description logics, ontologies.

Agradecimentos

O presente trabalho vem sendo produzido há seis anos desde quando comecei minha iniciação científica na área. Este doutorado foi financiado pela Fundação de Amparo a Pesquisa do Estado de São Paulo (FAPESP). Um trabalho com tamanha duração só foi possível com apoio de muitos que gostaria de agradecer aqui.

Raros são os casos de relação orientador orientado que duram tanto e de forma tão saudável quanto a que tive com minha orientadora. Atribuo a essa duradoura parceria com a professora Renata Wassermann o sucesso no desenvolvimento desta tese.

Outros professores do grupo de Lógica, Inteligência Artificial e Métodos Formais (LIAMF) do Instituto de Matemática e Estatística (IME) da Universidade de São Paulo (USP) estiveram presentes e participaram de uma forma ou de outra desta tese. O professor Marcelo Finger e o professor Flávio Soares me ajudaram imensamente participando da minha qualificação por duas vezes e dando ricas sugestões em ambas ocasiões. A professora Leliane Nunes Barros sempre se esforçou em me manter motivado o que em algumas ocasiões foi essencial. O professor Fabio Cozman, apesar de formalmente não fazer parte do grupo, acompanhou de perto minha trajetória e esteve sempre presente aos meus seminários desde o começo de minha iniciação científica.

Devo imensamente àqueles que me receberam no período que estive na Grécia e na França. A experiência de morar em um país de língua estrangeira é tão desafiadora quanto enriquecedora. Sem a ajuda de Giorgos Flouris, do professor Grigoris Antoniou e de Maria Moutsaki, a viagem não teria acontecido e eu não teria tipo essa experiência tão rica. Na França fui recebido com muitíssimo carinho pelo professor Andreas Herzig a quem serei eternamente grato.

Minha escolha pela carreira acadêmica foi certamente influenciada pelos professores que me educaram no começo do meu trajeto pela universidade, especialmente o professor Henrique Flemming, a professora Alinka Lépine, o professor Pedro Tonelli e o professor Manuel Garcia que foram inspirações para seguir esse caminho. O professor Marcos Gubitoso e a professora Cristina Fernandes foram muito importantes na minha decisão de ir para área de ciência da computação e permanecer nela respectivamente.

A Universidade de São Paulo foi minha casa durante esse tempo e gostaria de agradecer àqueles que a tornaram um lar para mim. Primeiramente aos colegas do curso ciências moleculares onde comecei essa trajetória. Estou certo de que sem eles eu não teria chegado até o fim do curso e muito menos ao fim do doutorado.

Acredito que só viveram plenamente o espaço da universidade aqueles que participaram de algum grupo de extensão. Grupos como o ALFA USP, o ComerAtivamente, a ITCP USP e tantos outros dão um sentido a universidade sem o qual não consigo mais concebê-la. Me sinto muito orgulhoso de ter participado do ALFA USP e do ComerAtivamente e espero que outros que passarem por essa universidade tenham o prazer de conhecer esses ou outros grupos de extensão e assim compreender esse sentido do qual me refiro.

Costumo dizer que sou um amaldiçoado por ter nascido em uma cidade pequena o suficiente para ter criado laços profundos com meus vizinhos, mas perto suficiente da metrópole para que ninguém se contentasse em seguir a vida no interior. Até hoje, morar em São Paulo tem sido um sacrifício para mim, sacrifício muito amenizado por aqueles que moraram comigo. Muitos estudantes como eu vem do interior para morar na capital com outros estudantes e para muitos a experiência de morar com outros é sacrificante. Esse não foi o meu caso. As pessoas com quem morei muito me ajudaram a encarar a metrópole. Tudo que aprendi com essas pessoas não caberia

nessas páginas de agradecimentos.

Vinhedo, porém, nunca deixou de ser meu lugar de refúgio da metrópole. Apesar de terem se espalhado pelo mundo, meus velhos amigos sempre arrumaram algum tempo para voltar a Vinhedo onde podíamos esquecer por um fim de semana ou um feriado da correria da metrópole.

Minha família esteve sempre ao meu lado. Meus pais receberam com dignidade o mundo deixado por meus avós e espero estar sendo digno do mundo que me foi deixado por eles, assim como espero que meus futuros filhos possam ser dignos do mundo que eu hei de construir para eles. Meus irmãos, além de terem sido grandes amigos, me serviram de inspiração em diversos momentos.

Por fim, quem mais me apoiou nos últimos cinco anos foi minha parceira, amiga e namorada. Agradecê-la em primeira pessoa do singular seria diminuir seu papel em tudo que construímos nesses anos. Mais honesto seria agradecermos (eu e ela) a todos aqueles citados anteriormente.

São Paulo, Agosto de 2010

Sumário

Conteúdo **vii**

Lista de Figuras **xi**

Lista de Tabelas **xiii**

1 Introdução **3**
1.1 Crenças Implícitas e Explícitas . . . 5
1.2 Ontologias e Web-semântica . . . 6
1.3 Publicações . . . 6
1.4 Organização da Tese . . . 7
1.5 Notação . . . 9

2 Lógica Proposicional Clássica **11**
2.1 Linguagem e Semântica da LPC . . . 11
2.2 Tableaux para LPC . . . 13
2.3 Propriedades da LPC . . . 14
2.4 Resumo . . . 15

3 Revisão de Crenças **17**
3.1 Sistemas de Crença . . . 17
3.2 Teoria AGM . . . 19
3.2.1 Suposições AGM . . . 20
3.2.2 Contração AGM . . . 20
3.2.3 Revisão AGM . . . 26
3.3 Bases de Crenças . . . 27
3.3.1 Contração . . . 28
3.3.2 Revisão . . . 31
3.3.3 Semi-Revisão . . . 33

4 Lógicas de Descrição **35**
4.1 Linguagem, Semântica e Tableaux . . . 35
4.1.1 Linguagem . . . 36
4.1.2 Semântica . . . 37
4.1.3 Tableaux . . . 39
4.1.4 Consistência . . . 42
4.1.5 Modelagem . . . 43
4.2 Extensões de $\mathcal{ALC}$ . . . 44
4.3 OWL . . . 46
4.4 Relação com Outros Formalismos . . . 48
4.4.1 Relação com a Lógica de Primeira Ordem . . . 48
4.4.2 Relação com Lógicas Modais . . . 49

5 Lógicas não Clássicas **53**
5.1 Compatibilidade AGM . . . 53
5.2 Reticulados como Heurísticas . . . 54
5.3 Propriedades Lógicas . . . 55
5.4 Algumas Lógicas não Clássicas . . . 59
5.4.1 Lógicas que Satisfazem as Suposições AGM . . . 59
5.4.2 Lógica de Horn . . . 60
5.4.3 Lógica Intuicionista . . . 61
5.4.4 Lógica Temporal Linear (LTL) . . . 62
5.4.5 Lógicas de Descrição . . . 63
5.5 Conclusão . . . 68

6 Contração AGM em Lógicas não Clássicas **69**
6.1 Recuperação e Relevância em Lógicas não Clássicas . . . 69
6.1.1 Relevância e Racionalidade AGM . . . 71
6.1.2 Relevância e Partial Meet . . . 72
6.1.3 Relevância e Compatibilidade . . . 74
6.2 Conclusão . . . 75

7 Revisão AGM em Lógicas sem Negação **77**
7.1 Construção . . . 77
7.2 Postulados e Teoremas de Representação . . . 78
7.3 Conclusão . . . 79

8 Bases de Crenças em Lógicas sem Negação **81**
8.1 Revisão Externa sem Negação . . . 81
8.1.1 Revisão Externa Kernel sem Negação . . . 83
8.1.2 Revisão Externa Partial Meet sem Negação . . . 86
8.2 Revisão Interna sem Negação . . . 88
8.2.1 Revisão Interna Partial Meet sem Negação (RIPM) . . . 88
8.2.2 Revisão Interna Kernel sem Negação (RIK) . . . 89
8.3 Conclusão . . . 90

9 Implementação das Operações **91**
9.1 Algoritmos Glass-Box e Black-Box . . . 91
9.2 Algoritmo de Reiter . . . 93
9.3 Relações entre o Conjunto Kernel e o Conjunto Resíduo . . . 94
9.4 Implementação . . . 95
9.5 Conclusão . . . 97

10 Conclusões e Trabalhos Futuros **103**
10.1 Conclusões . . . 103
10.2 Trabalhos Futuros . . . 104

A Graus de Recuperação em Bases de Crenças **105**
A.1 Contração em Bases sem Inclusão . . . 106
A.2 Adições Mínimas . . . 107
A.3 Graus de Inclusão e Recuperação . . . 108
A.4 Conclusão . . . 108

B Provas **109**
B.1 Provas do Capítulo 5 109
B.1.1 Relações entre propriedades lógicas 109
B.1.2 Lógica de Horn 110
B.1.3 Lógica intuicionista 110
B.1.4 Lógicas de descrição 111
B.2 Provas do Capítulo 6 112
B.3 Provas do Capítulo 7 114
B.4 Provas do Capítulo 8 116
B.4.1 Relações entre postulados 116
B.4.2 Revisão externa sem negação 117
B.4.3 Revisão interna sem negação 119
B.5 Provas do Capítulo 9 121

Referências Bibliográficas **123**

Referências Bibliográficas **123**

Índice Remissivo **129**

Índice Remissivo **130**

Lista de Figuras

1.1 Capítulos 8

2.1 Tableaux para LPC 13

3.1 Diagrama do exemplo 3.2. 21
3.2 Diagrama do exemplo 3.3. 22
3.3 Diagrama do exemplo 3.4. 22
3.4 Diagrama do exemplo 3.5. 23

4.1 Extensões de $\mathcal{ALC}$. 46
4.2 Complexidade das LDs sem axiomas GCI. 46
4.3 Complexidade das LDs com axiomas GCI. 47

5.1 Exemplo de lógica não decomponível [Flo06]. 55
5.2 Sub-reticulados que não ocorrem em lógicas distributivas. 57
5.3 Lógica decomponível, finita e não-distributiva. 58
5.4 Lógica booleana e não compacta. 59
5.5 Lógica de Horn com assinatura $\mathbb{P} = \{p, q\}$ 61
5.6 Reticulado de Rieger-Nishmura 62
5.7 Topologia das propriedades das lógicas. 67

8.1 Seções 82

9.1 Visão geral do Protégé com o plug-in para revisão. 96
9.2 Opções do plug-in para contração. 98
9.3 Opções do plug-in para revisão. 99
9.4 Elementos do conjunto kernel. 100
9.5 Elementos do conjunto resíduo. 101
9.6 O conceito *Saudável* antes da revisão. 102
9.7 O conceito *Saudável* depois da revisão. 102

B.1 A negação em lógicas finitas implica na decomponibilidade. 110

Lista de Tabelas

5.1 Propriedades das lógicas 66

6.1 Racionalidade AGM 72
6.2 Partial meet 74
6.3 Compatibilidade 75
6.4 Resumo dos resultados da Seção 6.1 76

8.1 Revisão em bases de crenças sem negação 90

Lista de Algoritmos

4.1 Tableaux para $\mathcal{ALC}$ 40
9.1 Algoritmo para achar um elemento do kernel. 92
9.2 Algoritmo glass-box para $\mathcal{ALC}$ 92
9.3 Algoritmo de Reiter para computar cortes mínimos em uma classe de conjuntos. . 93
9.4 Algoritmo para computar os demais elementos do kernel. 94

Capítulo 1

Introdução

O tema central desse trabalho é revisão de crenças em lógicas não clássicas e, em particular, em lógicas de descrição. Considere o seguinte exemplo motivador:

> **Exemplo 1.1:** Desde pequeno João acreditava que alimentos ricos em gordura como margarina e manteiga não são bons para saúde. Em uma aula de biologia ele aprendeu que existem diversos tipos de gordura. A gordura trans presente na margarina é um desses tipos de gordura e é especialmente ruim para a saúde.
>
> Mais tarde lendo uma revista João descobre a dieta do doutor Atkins. Segundo o Doutor Atkins, tanto a margarina quanto a manteiga podem ser consumidos a vontade, pois não são nocivas a saúde.

Ao lermos o exemplo acima atribuímos crenças ao João. No primeiro momento, por exemplo, atribuímos a ele a crença de que manteiga faz mal a saúde.

A área de revisão de crenças estuda como agentes mudam suas crenças ao receberem novas informações, por exemplo, quando João conheceu a dieta do Dr. Atkins ele mudou suas crenças sobre comidas saudáveis.

Em inteligência artificial, uma *crença* é uma determinada relação entre um agente (e.g.: "João") e uma proposição (e.g.: "alimentos ricos em gordura não são bons para saúde"). Um *agente*, por sua vez, é qualquer entidade capaz de perceber o mundo e atuar nele [RN03].

Segundo Russell e Norvig [RN03], um *agente racional* é aquele que a cada seqüência de percepções seleciona uma ação que acredita vir a maximizar sua medida de desempenho. Ou seja, ele escolhe ações que ele acredita que irão aproximá-lo de seus objetivos.

A definição de Russell e Norvig de agente racional se aproxima bastante da definição de Dennett [Den87] de *sistema intencional*. Para Dennett, um sistema é intencional se ele pode ser explicado e predito com base em atribuições a ele de crenças e desejos. Se atribuímos a João o desejo de se manter saudável e assumimos que ele acredita que alimentos ricos em gordura não são saudáveis e que a manteiga é rica em gordura, podemos prever que João irá evitar o consumo excessivo de manteiga.

Segundo Dennett, a intencionalidade não é uma característica intrínseca do sistema. O que determina se um sistema é intencional é a forma usada para explicar suas ações.

Seguindo essa abordagem identificamos uma primeira aplicação do estudo da revisão de crenças. A teoria de revisão de crenças deve nos dar uma resposta a seguinte pergunta: O que esperar de agentes racionais quando eles mudam suas crenças?

Suponha, por exemplo, que João acreditava que "Brasília é a capital do Brasil". Ao conhecer a dieta do Dr. Atkins ele pode mudar suas crenças sobre a manteiga ou a margarina, mas seria natural esperar que ele não mude sua crença sobre qual é a capital do país.

A totalidade das crenças que podem ser atribuídas a ele em um determinado momento é chamada de *estado epistêmico* de um agente. Em revisão de crenças são definidas operações em estados epistêmicos. No exemplo acima primeiramente o estado epistêmico de João contém certas crenças

sobre alimentos saudáveis. Quando João aprende sobre gordura trans atribuímos uma nova crença a ele. Ao fazermos isso aplicamos uma operação chamada *expansão* ao estado epistêmico de João. No momento seguinte, quando João conhece a dieta do Dr. Atkins, ele revisa suas crenças. A *revisão* é uma operação que acrescenta novas crenças a um estado epistêmico de forma consistente.

A teoria de revisão de crenças define critérios de racionalidade para operações em estados epistêmicos. Ou seja, critérios que um agente racional idealizado deve satisfazer. A minimalidade da mudança é um exemplo de critério de racionalidade: uma operação deve alterar o estado epistêmico de um agente de forma mínima.

Além da revisão e da expansão, a contração é outra operação importante em estados epistêmicos. A contração consiste na remoção de uma crença do estado epistêmico do agente.

O trabalho mais influente da área de revisão de crenças [AGM85] define um conjunto de postulados de racionalidade para cada uma das principais operações em estados epistêmicos. Esses conjunto de postulados são conhecidos como postulados AGM em homenagem aos autores desse trabalho.

Em inteligência artificial, porém, não estamos apenas interessados em estudar agentes racionais. Buscamos também desenvolver agentes (sistemas) que se comportem de maneira racional. Diversas sub-áreas de inteligência artificial buscam desenvolver tais sistemas. Nosso trabalho se inclui na área de *representação de conhecimento*.

A suposição principal dessa área é a seguinte:

> "Qualquer processo inteligente mecanicamente encarnado será composto de ingredientes que 1) nós como observadores externos tomamos como representação proposicional do conhecimento que o processo exibe como um todo 2) independente dessa atribuição semântica externa, desempenham um papel formal casual, mas essencial na concepção do comportamento manifesto por aquele conhecimento."

Essa afirmação de Smith [Smi82] é conhecida como *hipótese da representação do conhecimento*. Segundo essa hipótese, todo sistema inteligente possui uma estrutura interna que representa explicitamente seu estado epistêmico e tal estrutura, chamada de *base de conhecimento*, deve ser essencial na seleção das ações do sistema.

Identificamos, então, um segundo uso da teoria de revisão de crenças dentro da área de inteligência artificial: desenvolver sistemas baseados em conhecimento capazes de mudar suas bases de conhecimento racionalmente ao receberem novas informações. O papel da teoria de revisão de crenças, nesse caso, seria o de computar as operações de revisão e contração na base de conhecimento do agente.

A literatura de revisão de crenças é repleta de construções para essas operações [Gro88, Rot92, Gär88]. No próprio artigo do trio AGM são apresentadas formas de construir revisão e contração. O principal resultado de [AGM85] prova que ambas construções são totalmente caracterizadas pelos postulados AGM. Os teoremas que provam essas caracterizações são centrais na teoria de revisão de crenças e são chamados de teoremas de representação.

Se o trabalho do trio AGM pode ser considerado o mais influente na área de revisão de crenças, o trabalho de Brachman e Levesque [LB87] pode ser considerado pelo menos um dos mais influentes na área de representação de conhecimento. Nesse trabalho os autores argumentam por meio de exemplos concretos que existe um balanço entre a expressividade e a complexidade computacional de um formalismo. Portanto, para cada problema devemos tentar usar o formalismo mais adequado.

Nebel em sua tese [Neb90a] afirma que existem pelo menos dois importantes critérios de adequação que um formalismo deve satisfazer. O primeiro é a *adequação epistêmica*: um formalismo deve ser capaz de expressar tudo aquilo que for necessário para resolver o problema que se propõe. O segundo é a *adequação heurística*: o formalismo deve ser apropriado para ser usado internamente pelo sistema. Um exemplo importante desse segundo tipo de adequação é o tempo levado pelo agente para fazer inferências. Um agente que utiliza um formalismo muito expressivo pode levar muito tempo para fazer uma inferência crucial e, portanto, não ser adequado.

As lógicas de descrição são uma família de formalismos usados para representar conhecimento conceitual. A grande vantagem das lógicas de descrição com relação a seus antecessores (sistemas de frames [Min81] e redes semânticas [Qui67]) é possuírem semântica bem definida. Diversas lógicas de descrição com diferentes expressividades e complexidades foram estudadas na literatura e um dos principais desafios da área foi o de encontrar os formalismos mais expressivos em cada classe de complexidade.

Para os propósitos dessa tese, uma lógica é simplesmente um par $\langle \mathscr{L}, Cn \rangle$ em que $\mathscr{L}$ é a linguagem da lógica e Cn é seu operador de consequência. Ou seja, $\mathscr{L}$ é o conjunto de todas as sentenças admitidas na lógica e Cn é uma função que leva um conjunto de sentenças $B \in 2^{\mathscr{L}}$ ao conjunto de suas consequências $Cn(B)$.

Muitos outros aspectos de uma lógica são relevantes, mas de maneira abstrata assumimos apenas que uma lógica seja um par $\langle \mathscr{L}, Cn \rangle$. Dedicamos o Capítulo 2 à mais estudada das lógicas: a lógica proposicional clássica. Nesse capítulo abordamos diversos aspectos dessa lógica. As demais lógicas discutidas nesse trabalho serão sempre comparadas à lógica clássica.

A teoria AGM não se aplica a qualquer lógica. Grande parte da literatura da área de revisão de crenças assume que a lógica usada satisfaz certas suposições que chamamos de *suposições AGM*. Essas suposições se aplicam à lógica clássica, mas não se aplicam por exemplo a diversas lógicas de descrição. Trabalhos recentes [Flo06, FPA05, FPA06] mostraram as dificuldades de se aplicar a teoria AGM a lógicas de descrição. Esses trabalhos mostram que a contração AGM não é nem mesmo definível em determinadas lógicas.

O objetivo central desta tese é aplicar os resultados da teoria de revisão de crenças a lógicas que não satisfazem as suposições AGM. Nossa abordagem consiste em estudar propriedades lógicas que garantem determinados resultados de revisão de crenças. Seguindo a tradição da área, não faremos distinção entre o termo crença e o termo conhecimento por não ser o foco do trabalho.

Estudaremos tanto revisão em conjunto de crenças quanto revisão em bases de crenças. A seção seguinte apresenta a diferença entre as duas de maneira informal e introdutória.

Em seguida apresentamos uma aplicação importante do estudo de revisão de crenças em lógicas de descrição: a web-semântica. Concluímos o capítulo apresentando a organização da tese e a notação que usaremos no restante dela.

1.1 Crenças Implícitas e Explícitas

Em [AGM85] os autores representam um estado epistêmico como um conjunto de sentenças K fechado por consequência lógica ($K = Cn(K)$). Esta abordagem não distingue o conhecimento explícito do conhecimento implícito do agente.

Em uma outra abordagem chamada de revisão em bases de crenças [Han99, Fuh97, Was00b], as crenças contidas no estado epistêmico de um agente são separadas em crenças implícitas e crenças explícitas. Nessa abordagem representamos as crenças explícitas de um agente por um conjunto arbitrário de sentenças $B \in 2^{\mathscr{L}}$. As consequências de B, apesar de fazerem parte do estado epistêmico do agente, possuem um status inferior.

Considere o exemplo tirado de [Han99] que ilustra a diferença entre as abordagens:

> **Exemplo 1.2:** Eu acredito que Paris é a capital da França (α). Também acredito que tem leite na geladeira (β). Logo, eu acredito que Paris é a capital da França se e somente se tem leite na geladeira ($\alpha \leftrightarrow \beta$). Abro minha geladeira me vejo obrigado a trocar minha crença em β por $\neg\beta$. Não posso então manter minhas crenças em α e em $\alpha \leftrightarrow \beta$ ao mesmo tempo.
>
> *Abordagem usando conjuntos de crenças:* Tanto α quanto $\alpha \leftrightarrow \beta$ são elementos do conjunto de crenças. Quando abro minha geladeira preciso escolher entre reter a crença em α ou reter minha crença em $\alpha \leftrightarrow \beta$. A remoção de $\alpha \leftrightarrow \beta$ não segue automaticamente. Ela poderia ser garantida por um mecanismo de seleção.

Abordagem usando bases de crenças: A crença em $\alpha \leftrightarrow \beta$ é meramente derivada das demais. Quando β é removido $\alpha \leftrightarrow \beta$ automaticamente desaparece.

Ambas as abordagens possuem fortes atrativos. Na abordagem AGM estados epistêmicos equivalentes são tratados de forma equivalente. Ou seja, essa abordagem abstrai a *forma* como são representadas as crenças do agente. A abordagem de base de crenças, por sua vez, é mais expressiva e mais interessante do ponto de vista computacional.

Do ponto de vista filosófico existem autores que defendem a distinção entre crenças implícitas e explicitas [Har86] como existem aqueles que defendem o contrário [Sta84]. Nesse trabalho exploraremos ambas abordagens. No Capítulo 6 estudamos revisão em conjuntos de crenças e reservamos o Capítulo 8 para as bases de crenças.

1.2 Ontologias e Web-semântica

Nesse trabalho abordamos a aplicação da teoria da revisão de crenças em lógicas não clássicas. Em particular estamos interessados nas lógicas de descrição. O interesse principal nas lógicas de descrição está na sua capacidade de representar ontologias.

Uma definição bastante aceita de ontologia é a de Gruber [Gru93]:

> "uma ontologia é uma especificação formal e distribuída de uma conceitualização de um domínio"

Ou seja, uma ontologia explicita conceitos e relações abstratas por meio de uma linguagem formal.

Em computação, o estudo de ontologias ganhou destaque com o desenvolvimento da *web-semântica*. A web-semântica [BLHL01] é descrita como uma possível evolução da web em que as informações disponíveis seriam compreensíveis tanto por seres humanos (como a web de hoje) quanto por máquinas.

O papel das ontologias na web-semântica seria o de fornecer um vocabulário comum entre agentes na web. Ontologias seriam responsáveis por declarar os termos usados por cada agente e explicitar como esses termos se relacionam semanticamente. Desta forma, seria possível uma comunicação mais rica entre agentes na web.

Lógicas de descrição são uma família de formalismos (lógicas) usados para representar conhecimento conceitual (ontologias). O estudo das lógicas de descrição foi marcado pela busca de um balanço satisfatório entre expressividade e complexidade computacional. Além disso, o fato de possuírem semântica bem definida fez das lógicas de descrição as candidatas ideais para representar ontologias na web. A linguagem padrão para representar ontologias na web (OWL) tem como base determinadas lógicas de descrição. No Capítulo 4 apresentaremos as lógicas de descrição em mais detalhes.

1.3 Publicações

Parte dos resultados expostos nesta tese foram publicados durante os últimos quatro anos em workshops, conferências e revistas. A seguir listamos em ordem cronológica destas publicações:

1. "First Steps Towards Revising Ontologies" [RW06] escrito com Renata Wassermann e apresentado no segundo Workshop on Ontologies and their Aplications (WONTO 2006).

2. "Classifying Ontologies" [KPTR$^+$06] escrito em parceria com Renata Wassermann, Fábio Kepler, Christian Paz-Trillo, Joselyto Riani, Karina Valdivia-Delgado e Leliane Nunes e também apresentado no WONTO 2006.

3. "Base Revision in Description Logics - Preliminary Results" [RW07], escrito com Renata Wassermann e apresentado no Internacional Workshop on Ontology Dynamics 2007 (IWOD 2007).

4. "Degrees of Recovery and Inclusion in Belief Base Dynamics" [RW08a] escrito com Renata Wassermann e apresentado no International Workshop on Non-Monotonic Reasoning 2008 (NMR 2008).

5. "The Ontology Reviser Plug-In for Protégé" [RW08b] escrito com Renata Wassermann e apresentado no terceiro Workshop on Ontologies and their Aplications (WONTO 2008)

6. "On the Relation Between Remainder Sets and Kernels" [RW08c] escrito com Renata Wassermann e apresentado na XV Encontro Brasileiro de Lógica em Paraty (EBL 2008).

7. "AGM Revision in Description Logics" [RW09a] escrito com Renata Wassermann e apresentado no primeiro Workshop on Automated Reasoning about Context and Ontology Evolution (ARCOE 2009)

8. "Base Revision for Ontology Debugging" [RW09b] escrito com Renata Wassermann e publicado no Journal of Logic and Computation (JLC 2009).

9. "Contraction on the Semantic Web: On the Role of Relevance and Recovery" [RWA$^+$09] escrito em parceria com Renata Wassermann, Giorgos Flouris, Grigoris Antoniou e Jeff Pan e apresentado no International Workshop on Ontology Dynamics 2009 (IWOD 2009).

10. "More about AGM Revision in Description Logics" [RW10] escrito com Renata Wassermann e apresentado no workshop em segundo Workshop on Automated Reasoning about Context and Ontology Evolution (ARCOE 2010).

1.4 Organização da Tese

Dividimos a tese em duas partes. A primeira composta pelos Capítulos de 2 a 4 é uma introdução aos temas abordados. A segunda parte, formada pelos demais capítulos, contém nossas contribuições.

O Capítulo 2 apresenta a lógica proposicional clássica. O intuito principal desse capítulo é mostrar aspectos que devem ser analisados ao se apresentar uma lógica: a linguagem, a semântica, o método de inferência, as propriedades etc. Nos remeteremos a esse capítulo sempre que formos apresentar alguma lógica.

O Capítulo 3 apresenta a teoria de revisão de crenças. O capítulo trata tanto da revisão em conjuntos de crenças [AGM85, Gär88] como em bases de crenças [Han99, Was00b]. Em ambos os casos são apresentadas construções para contração e para revisão e são mostrados os principais teoremas de representação.

O Capítulo 4 apresenta as lógicas de descrição. Como existem diversas lógicas de descrição são usadas a lógica bem conhecida $\mathcal{ALC}$ como exemplo para definir a linguagem, a semântica e um método de inferência. Em seguida são definidas diversas lógicas da família $\mathcal{ALC}$ e $\mathcal{S}$. Mostramos como as lógicas de descrição estão relacionadas com a linguagem para representação de ontologias na web (OWL). Por fim, mostramos a relação entre lógicas de descrição e outros formalismos como a lógica de primeira ordem e algumas lógicas modais.

Começamos a segunda parte da tese pelo Capítulo 5 em que mostramos propriedades que o operador Cn pode satisfazer e como essas propriedades se relacionam. Começamos esse capítulo motivando o estudo da teoria AGM em lógicas não clássicas. Apresentamos o trabalho de Flouris [Flo06] em que é definida a compatibilidade AGM. Estendemos o trabalho de Flouris mostrando uma série de lógicas que não são compatíveis com os postulados AGM. Essa lista de lógicas contém, além de diversas lógicas de descrição, a lógica intuicionista e a lógica de Horn.

No Capítulo 6 começamos o estudo da revisão AGM em lógicas não clássicas. Como os postulados AGM para contração não são aplicáveis a diversas lógicas, propomos um conjunto de postulados para substituí-los. Esse novo conjunto de postulados é compatível com uma gama muito maior de lógicas que inclui grande parte das lógicas apresentadas no capítulo anterior.

Comparamos esse novo conjunto de postulados aos postulados AGM e mostramos, em particular, que ele caracteriza melhor a contração partial meet. Além disso, mostramos que ele não é equivalente aos postulados AGM em determinadas lógicas. Concluímos que nessas lógicas em que a noção de minimalidade não é equivalente a noção de recuperabilidade.

Ainda nesse capítulo mostramos uma construção para a revisão AGM que não depende da definição de negação. Essa revisão, portanto, pode ser aplicada inclusive em lógicas que não possuem negação bem definida (i.e. definida para toda sentença da linguagem) como a lógica de Horn e as lógicas de descrição. Apresentamos um conjunto de postulados que caracterizam essa construção e provamos que esse conjunto de postulados é equivalente aos postulados AGM caso a lógica subjacente satisfaça as suposições AGM.

No Capítulo 8 passamos a estudar revisão em bases de crenças. Apresentamos diversas construções (seis ao todo) que não dependem da definição da negação. Para cada uma dessas construções provamos o teorema de representação que a relaciona com o respectivo conjunto de postulados.

O diagrama da figura 1.1 resume a organização dos Capítulos 6 e 8. Nesses capítulos tratamos tanto da teoria de revisão em conjuntos (Capítulo 6) quanto em bases (Capítulo 8) de crenças. Em conjuntos de crenças tratamos tanto da contração (Seção 6.1) quanto da revisão (Seção 7). Sobre bases de crenças não tratamos da contração que foi suficientemente estudada em [HW02], mas tratamos tanto da revisão externa (Seção 8.1) quanto interna (Seção 8.2). Em ambos os casos estudamos tanto a revisão kernel e quanto a revisão partial meet.

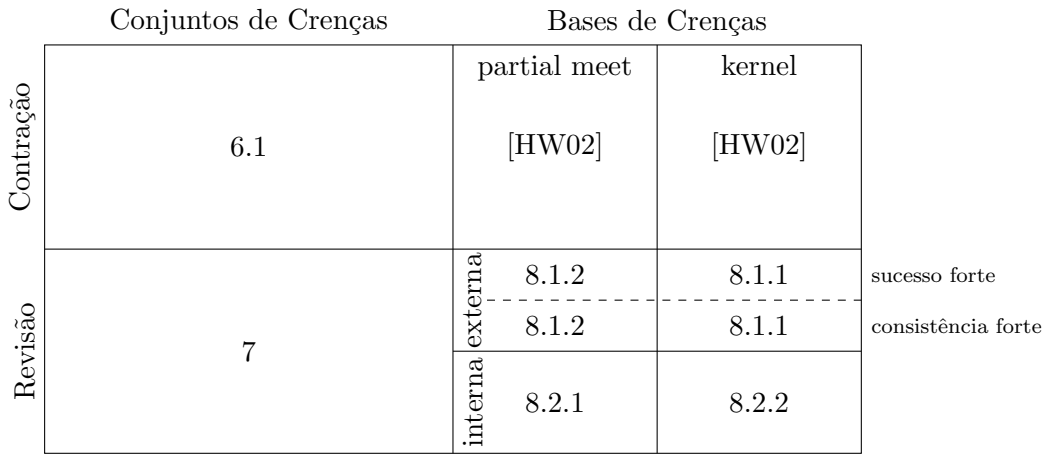


Figura 1.1: Capítulos

No Capítulo 9 mostramos como usar algoritmos já conhecidos da literatura de depuração de ontologias para implementar as construções para contração e revisão em bases de crenças. Nossa principal contribuição nesse capítulo é estabelecer uma ponte entre os algoritmos estudados na área de depuração de ontologias e a teoria da revisão em bases de crenças. Mostramos que por um lado, é possível usar esses algoritmos para implementar as construções em bases de crenças e, por outro, é possível caracterizar os algoritmos usando os teoremas de representação da teoria de bases de crenças. Essa caracterização é útil para abordar o problema de um ponto de vista mais abstrato e, assim, aprofundar a visão do problema.

No Capítulo 10 concluímos e resumimos nossas contribuições enumerando os artigos publicados durante o desenvolvimento dessa tese.

No apêndice A mostramos que em bases de crenças o relaxamento do postulado da inclusão leva a diferentes graus de recuperação. Esse trabalho foi incluído por se relacionar indiretamente com o tema da tese e ter sido desenvolvido no mesmo período.

O apêndice B, por fim, contém as demonstrações.

1.5 Notação

No que segue usaremos as letras gregas minúsculas (α, β ...) para denotar sentenças em uma linguagem $\mathscr{L}$. Letras maiúsculas (A, B ...) são usadas em geral para representar conjuntos de sentenças. Em algumas situações usaremos essas mesmas letras para representar conceitos em uma lógica de descrição. Será sempre claro pelo contexto se o símbolo representa um conjunto de sentenças ou um conceito. Reservamos a letra maiúscula K para conjuntos logicamente fechados. Letras minúsculas (a, b ...) representam variáveis proposicionais na lógica clássica e nas lógicas modais ou indivíduos na lógicas de descrição. O símbolo $\subseteq$ representa a inclusão de conjuntos e o símbolo $\subset$ será usado exclusivamente para a inclusão própria de conjuntos. Os demais símbolos serão apresentados ao longo do trabalho.

Capítulo 2

Lógica Proposicional Clássica

No presente trabalho lidaremos com diversas lógicas. Nossa abordagem consiste em considerar uma lógica de um ponto de vista abstrato como uma dupla $\langle \mathscr{L}, Cn \rangle$ que satisfaz determinadas propriedades e mostrar como resultados de revisão de crenças se aplicam a essas lógicas. Nesta seção apresentaremos a lógica proposicional clássica (LPC). Ao fazer isso enfatizaremos os aspectos que devem ser levados em conta ao apresentar outras lógicas.

2.1 Linguagem e Semântica da LPC

O primeiro passo para definir uma lógica é definir o conjunto de *sentenças* ou de *fórmulas bem formadas* da lógica. Chamaremos esse conjunto de *linguagem* da lógica e o representaremos pelo símbolo $\mathscr{L}$. A linguagem $\mathscr{L}$ é definida por uma gramática a partir de um ou mais conjuntos de símbolos atômicos chamados de *assinatura* da lógica. A assinatura da LPC é um conjunto enumerável $\mathbb{P}$ cujos elementos são chamados de *variáveis proposicionais*.

A linguagem $\mathscr{L}_{LPC}$ da lógica proposicional é definida a partir de $\mathbb{P}$ pela seguinte gramática descrita em BNF [1]:

$$\alpha := \ p \mid \bot \mid \top \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \alpha \rightarrow \alpha \mid \neg\alpha$$

Essa gramática abrevia a seguinte definição formal:

O conjunto $\mathscr{L}_{LPC}$ é o menor conjunto que satisfaz as seguintes regras:

1. $\mathbb{P} \cup \{\bot, \top\} \subseteq \mathscr{L}_{LPC}$
2. Se $\alpha, \beta \in \mathscr{L}_{LPC}$ então $\alpha \wedge \beta, \alpha \vee \beta, \alpha \rightarrow \beta, \neg\alpha \in \mathscr{L}_{LPC}$.

Uma vez definida a linguagem da lógica, devemos definir o *operador de consequência* $Cn : 2^{\mathscr{L}} \rightarrow 2^{\mathscr{L}}$ que devolve as sentenças que seguem logicamente de um determinado conjunto $A \in 2^{\mathscr{L}}$.

Antes de definirmos o operador Cn, porém, definiremos modelos em um lógica proposicional. Assumimos que existe uma classe de modelos associada à lógica e que cada sentença $\alpha \in \mathscr{L}$ está associada a um subconjunto que chamaremos de *modelos* de α. O conjunto de modelos de um conjunto de sentenças $B \in 2^{\mathscr{L}}$ é a intersecção dos modelos das sentenças em B. Dizemos que uma sentença (conjunto de sentenças) é *satisfatível* nessa classe de modelos se existe pelo menos um modelo para essa sentença (conjunto de sentenças). Dizemos que uma sentença é *válida* se todos os elementos da classe são modelos para ela. Dizemos que duas sentenças α e β são *equivalentes* se elas possuem os mesmos modelos.

Em lógica proposicional clássica, a classe de modelos é formada por todas as funções $v : \mathbb{P} \rightarrow \{0, 1\}$ chamadas *valorações*. Dizemos que uma valoração v satisfaz uma fórmula atômica p sse $v(p) = 1$ (escrevemos $\vDash_v p$). A relação de satisfatibilidade $\vDash_v$ é estendida às demais sentenças da linguagem de forma recursiva:

[1] Forma de Backus-Naur.

$\models_v \top$

$\not\models_v \bot$

$\models_v \neg\alpha$ sse $\not\models_v \alpha$

$\models_v \alpha \wedge \beta$ sse $\models_v \alpha$ e $\models_v \beta$

$\models_v \alpha \vee \beta$ sse $\models_v \alpha$ ou $\models_v \beta$

$\models_v \alpha \rightarrow \beta$ sse $\not\models_v \alpha$ ou $\models_v \beta$

Podemos definir a consequência semântica como:

$$\begin{aligned} Cn(A) &= \{\alpha \in \mathscr{L} : \text{o conjunto de modelos de } A \\ &\quad \text{está contido no conjunto de modelos de } \alpha\}. \end{aligned}$$

Operadores de consequência semântica Cn definidos a partir de modelos da forma como apresentamos acima satisfazem as seguintes propriedades para todo $A, B \in 2^{\mathscr{L}}$:

monotonicidade: se $A \subseteq B$ então $Cn(A) \subseteq Cn(B)$.

inclusão: $A \subseteq Cn(A)$.

idempotência: $Cn(A) = Cn(Cn(A))$.

Qualquer lógica $\langle \mathscr{L}, Cn \rangle$ cujo operador de consequência satisfaz essas três propriedades é chamada de lógica *tarskiana*.

A tarskianicidade segue de dois fatos triviais sobre modelos. Primeiro se $A \subseteq B$ então o conjunto de modelos de B deve estar contido no conjunto de modelos de A, visto que consideramos a intersecção dos modelos. Segundo, os modelos de $Cn(A)$ são os mesmos de A, pois as fórmulas em $Cn(A)$ são exatamente aquelas cujos modelos contém os modelos de A.

Da forma como definimos o operador Cn, duas sentenças α e β são equivalentes se e somente se $Cn(\alpha) = Cn(\beta)$.

A seguir vamos definir o operador de consequência da LPC que será denotado por C_{LPC} .

Assim, $\alpha \in C_{LPC}(B)$ sse $B \cup \{\neg\alpha\}$ é insatisfatível. Ou seja, o problema de decidir se uma sentença é consequência de um conjunto de sentenças pode ser reduzido ao problema da satisfatibilidade.

Exemplo 2.1: A sentença $p \vee \neg p$ é válida, pois, se $v(p) = 1$ então $\models_v \neg p \vee p$ e se $v(p) = 0$ então $\models_v \neg p$ e nesse caso $\models_v p \vee \neg p$.

Exemplo 2.2: $q \in C_{LPC}(\{p, p \rightarrow q\})$, pois, se $\models_v \{p, p \rightarrow q\}$ então $\models_v p$ e $\models_v p \rightarrow q$. Nesse caso $v(p) = 0$ ou $v(q) = 1$ e como $v(p) = 1$ temos que $v(q) = 1$.

Do ponto de vista computacional não basta apenas definir o operador de consequência, é preciso mostrar uma forma sistemática para decidir se uma sentença α é consequência de um conjunto B ($\alpha \in Cn(B)$). Em outras palavras, é necessário definir um *procedimento de prova* para uma lógica.

2.2 Tableaux para LPC

No restante deste capítulo mostraremos um procedimento de prova para LPC baseado no método de tableaux. Escolhemos o método de tableaux por ser versátil e ser adaptável a diversas lógicas: lógicas modais, lógicas de descrição, lógica intuicionista etc.

Antes de mostrar o método de tableaux vamos mostrar que toda sentença $\alpha \in \mathscr{L}_{LPC}$ é equivalente a uma sentença α' tal que o símbolo $\rightarrow$ não ocorre em α' e o símbolo $\neg$ ocorre apenas na frente de variáveis proposicionais. A sentença α' com essas características é chamada de *forma normal negada* (FNN) de α. Uma sentença pode ser posta na sua forma normal negada substituindo toda ocorrência de $\alpha \rightarrow \beta$ por $\neg\alpha \vee \beta$ e "empurrando a negação para dentro" usando as seguintes equivalências:

$$\begin{aligned} \neg\top &\equiv \bot \\ \neg\bot &\equiv \top \\ \neg\neg\alpha &\equiv \alpha \\ \neg(\alpha \wedge \beta) &\equiv \neg\alpha \vee \neg\beta \\ \neg(\alpha \vee \beta) &\equiv \neg\alpha \wedge \neg\beta \end{aligned}$$

As duas últimas equivalências são conhecidas como *leis de De Morgan*.

O método de tableaux que apresentaremos é uma adaptação do método apresentado em [Smu68]. Esse método recebe um conjunto finito $B \in 2^{\mathscr{L}}_{LPC}$ em que todas as fórmulas estão na forma normal negada e decide se B é satisfatível. A estrutura de dados que usaremos é um simples conjunto de conjuntos de sentenças chamados *ramos* . Um ramo B possui um *conflito* se B contém $\bot$ ou se para algum $p \in \mathbb{P}$ temos que $p, \neg p \in B$. Um ramo está *completo* ou *saturado* se não há mais regras aplicáveis a ele.

Começamos, então com um único ramo $\{B\}$ e aplicamos as regras do algoritmo 2.1 até que todos os ramos estejam ou saturados ou possuam um conflito. Se ao fim desse processo todos os ramos possuem conflitos dizemos que o tableau está *fechado* e o procedimento devolve `insatisfatível`. Caso contrário o procedimento devolve `satisfatível`.

regra $\rightarrow \wedge$:

condição: A contém uma fórmula $\alpha = \alpha_1 \wedge \alpha_2$, mas não contém α_1 ou não contém α_2

ação: $A := A \cup \{\alpha_1, \alpha_2\}$.

regra $\rightarrow \vee$:

condição: A contém $\beta = \beta_1 \vee \beta_2$, mas não contém β_1 nem β_2.

ação: cria o conjunto $A' := A \cup \{\beta_1\}$ e $A := A \cup \{\beta_2\}$

Figura 2.1: Tableaux para LPC

Exemplo 2.3: Vamos provar que q é consequência lógica de $\{p, p \rightarrow q\}$. Para tanto, temos que mostrar que $\{p, p \rightarrow q, \neg q\}$ é insatisfatível. Primeiro vamos passar todas as sentenças para forma normal negada. A única sentença que ainda não está em FNN é $p \rightarrow q$ que é equivalente a $\neg p \vee q$. Agora aplicamos as regras de tableaux para $\{p, \neg p \vee q, \neg q\}$.

$$\begin{aligned} A_0 &= \{p, \neg p \vee q, \neg q\} \\ A_1 &= A_0 \cup \{\neg p\} \qquad \rightarrow \vee \\ A_0 &= A_0 \cup \{q\} \qquad \rightarrow \vee \end{aligned}$$

Tanto A_0 quanto A_1 possuem conflitos e, portanto, $\{p, \neg p \vee q, \neg q\}$ é insatisfatível. Logo, $q \in Cn(\{p, p \rightarrow q\})$.

Sempre que apresentamos um método de prova há três perguntas que devemos procurar responder:

- Esse método é correto? Ou seja, sempre que o tableau fecha o conjunto é insatisfatível?
- Esse método é completo? Ou seja, para todo conjunto finito de sentenças insatisfatível o tableau fecha?
- Esse método termina? Ou seja, para qualquer entrada finita o algoritmo devolve alguma resposta em tempo finito?

Um procedimento de prova correto, completo e que termina é chamada de *método de decisão*. De fato o método apresentado aqui é um método de decisão para o problema da satisfatibilidade em LPC (SAT).

A correção se verifica notando que se uma regra $\rightarrow \wedge$ é aplicada a um conjunto satisfatível, o novo conjunto ainda é satisfatível e se uma regra $\rightarrow \vee$ é aplicada a um conjunto satisfatível, pelo menos um dos novos conjuntos é satisfatível. Assim, se todos os ramos possuem conflitos, ou seja todos ramos são insatisfatíveis, é porque o conjunto inicial era insatisfatível.

A completude se verifica usando uma *interpretação canônica* para B. Se o tableau possui um ramo saturado B', definimos uma interpretação v tal que $v(p) = 1$ para toda variável proposicional $p \in B'$ e $v(q) = 0$ para toda variável proposicional q tal que $\neg q \in B'$ e $v(r) = 0$ para as demais variáveis proposicional (na verdade podemos atribuir qualquer valor a essas variáveis). Não é difícil notar que essa interpretação satisfaz o conjunto B. Completamos a argumentação dizendo que se o tableau fosse aberto então B seria satisfatível, logo, se B não é satisfatível o tableau deve fechar.

O procedimento sempre termina. Mais precisamente, sabemos que o problema SAT é NP-completo. Ou seja, existe um procedimento polinomial para verificar uma dada solução (pertence a classe NP) e qualquer problema que possua essa característica pode ser reduzido ao problema SAT (NP-completo). O pertencimento a classe NP é trivial, pois é possível verificar em tempo polinomial se uma dada interpretação satisfaz uma fórmula α. A prova da NP-completude para o problema SAT é um resultado célebre da teoria da computação conhecido como teorema de Cook [Coo71].

2.3 Propriedades da LPC

Como enfatizamos diversas vezes nas seções anteriores, nossa abordagem consiste em estudar as lógicas de um ponto de vista abstrato considerando-as como pares $\langle \mathscr{L}, Cn \rangle$ que satisfazem algumas propriedades. Já mostramos que a LPC satisfaz monotonicidade, inclusão e idempotência, concluiremos esse capítulo mostrando mais algumas propriedades que ela satisfaz.

Uma propriedade importante que vale para LPC é chamada de *compacidade*. A compacidade, como veremos mais adiante é uma propriedade central para a revisão de crenças. Enunciamos a compacidade da seguinte forma:

compacidade: Uma lógica $\langle \mathscr{L}, Cn \rangle$ é dita compacta sse para todo $B \in 2^{\mathscr{L}}$ se $\alpha \in Cn(B)$ então existe $B' \subseteq B$ finito tal que $\alpha \in Cn(B')$.

Outra forma de enunciar essa propriedade é a seguinte: se todo subconjunto finito B' de B é satisfatível então B é satisfatível. Pela redução do problema da consequência lógica ao problema da satisfatibilidade temos que ambas versões são equivalentes.

Não pretendemos escrever com detalhes a demonstração da compacidade para LPC (para isso veja [Smu68]), mas esboçaremos as idéias principais, pois elas serão úteis mais a diante.

Uma forma de provar compacidade para LPC é através do *lema de Tuckey*:

Lema 2.4 (Tuckey [Smu68])**.** *Uma propriedade P tem* caráter finito *se para qualquer conjunto S, S satisfaz P sse todos seus subconjuntos finitos satisfazem P. Seja U um conjunto chamado de universo. Seja P uma propriedade de caráter finito. Se $S \subseteq U$ satisfaz P então S pode ser estendida a um conjunto maximal que tenha a propriedade P.*

No caso em que U não é enumerável, o lema de Tuckey é sabidamente equivalente ao axioma da escolha [Smu68].

Desse lema temos que se $A \in 2^{\mathscr{L}}$ é satisfatível então é possível estender A a um conjunto A' maximal tal que qualquer subconjunto finito de A' é satisfatível. Essa consequência do lema de Tuckey é conhecida como *lema de Lindembaum*.

Usando o lema de Lindembaum provamos que a partir de um conjunto A' maximal cujos subconjuntos finitos são satisfatíveis podemos construir uma interpretação para A. Para tanto basta atribui $v(\alpha) = 1$ para todo $\alpha \in A'$ e $v(\alpha) = 0$ para as demais sentenças de $\mathscr{L}$. Portanto, a compacidade da LPC segue do lema de Lindembaum que, por sua vez segue do lema de Tuckey.

Outra propriedade da LPC é a *regra da eliminação da dupla negação*:

$$\alpha \in Cn(\neg\neg\alpha)$$

Uma consequência importante dessa propriedade é a chamada *regra do terceiro excluído*.

$$\alpha \vee \neg\alpha \in Cn(\emptyset)$$

Ambas propriedades são rejeitadas na lógica intuicionista (veja a seção 5.4.3).

A última propriedade do operador C_{LPC} que vamos mostrar é a chamada de dedução:

dedução: Uma lógica $\langle \mathscr{L}, Cn \rangle$ satisfaz a *dedução* se vale que $\alpha \in Cn(B \cup \{\beta\})$ sse $\beta \to \alpha \in Cn(B)$.

De novo não entraremos em detalhes, mas essa demonstração segue direto da definição que demos para $\alpha \to \beta$ e para o operador C_{LPC}.

2.4 Resumo

Resumiremos os principais aspectos apresentados sobre a lógica proposicional clássica para servir de roteiro para as demais lógicas que apresentaremos nesse trabalho.

1. definimos a linguagem $\mathscr{L}_{LPC}$ usando uma gramática em BNF.
2. definimos a classe das valorações $v : \mathscr{L}_{LPC} \to \{0, 1\}$ como a classe dos modelos para LPC.
3. mostramos como verificar se uma valoração é modelo de uma sentença $\alpha \in \mathscr{L}_{LPC}$.
4. definimos o operador C_{LPC} a partir desses modelos.
5. apresentamos um método de decisão para o problema da satisfatibilidade.
6. mostramos algumas propriedades de $\langle L_{LPC}, C_{LPC} \rangle$: compacidade e dedução.

Capítulo 3

Revisão de Crenças

Neste capítulo introduziremos os principais conceitos e resultados da área de revisão de crenças. Revisão de crenças é a sub-área de representação de conhecimento que estuda a dinâmica dos estados epistêmicos de um agente.

Na próxima seção definiremos sistemas de crenças genéricos como em [Gär88]. Sistemas de crenças são compostos por uma forma de representar estados epistêmicos, um conjunto de atitudes epistêmicas e tipos de mudanças de crenças. Nas próximas seções apresentaremos dois sistemas de crenças específicos. Na seção 3.2 estudaremos um sistema de crenças em que estados epistêmicos são representado por conjuntos de sentenças logicamente fechados. Essa abordagem é chamada de teoria AGM em homenagem aos autores de [AGM85]. Na seção 3.3 trataremos de um sistema de crenças em que os estados epistêmicos são representados por conjuntos arbitrários de sentenças. Essa abordagem distingue crenças implícitas de crenças explícitas e é chamada de teoria das bases de crenças [Han99].

3.1 Sistemas de Crença

Nesta seção definiremos sistemas de crenças de forma genérica como apresentado por Gärdenfors em [Gär88].

O *estado epistêmico* de um agente representa o conjunto de tudo o que ele acredita e como essas crenças se relacionam em um determinado momento. Gärdenfors [Gär88] define um estado de crenças como uma representação idealizada do estado cognitivo de um agente em determinado momento. Alternativamente podemos entender o estado epistêmico como uma representação de todas crenças atribuídas a um agente em um determinado momento.

Nesse trabalho vamos considerar apenas modelos de estados de crenças baseados em conjuntos de sentenças lógicas. Não trataremos, por exemplo, de modelos de estados epistêmicos probabilísticos [Pea88].

Nas seções 3.2 e 3.3 exploramos duas formas de representar estados epistêmicos. Na primeira ele é representado como um conjunto de sentenças logicamente fechado (conjuntos de crenças). Na segunda é representado como um conjunto arbitrário de sentenças (bases de crenças).

Dado um modelo do estado de crenças, um agente pode ter uma série de *atitudes epistêmicas* frente a cada elemento desse modelo. Em um modelo probabilístico, por exemplo, o agente pode aceitar ou rejeitar uma sentença com uma certa probabilidade. Como veremos em mais detalhes na seção 3.2, partindo de estados epistêmicos modelados como conjuntos de crenças um agente pode ter três principais atitudes perante uma sentença: a sentença é aceita, rejeitada ou indeterminada. Partindo de um estado epistêmico modelado como uma base o agente tem quatro possíveis atitudes epistêmicas com relação a uma sentença: aceitar explicitamente, aceitar implicitamente, rejeitar[1] ou não ter opinião (indeterminada).

A área de revisão de crenças estuda a dinâmica dos estados epistêmicos de um agente. Ou seja, o estudo das mudanças que ocorrem nos estados epistêmicos de um agente. Tais mudanças

[1] Em lógicas fechadas por negação podemos também distinguir rejeição implícita da rejeição explícita.

correspondem a mudança de atitude perante um elemento do modelo. Segundo Gärdenfors [Gär88], essas mudanças deve ser motivadas por algum "disparador" externo ao agente chamado de *entrada epistêmica*. A forma dessa entrada é irrelevante, pode ser simplesmente uma frase ouvida pelo agente ou um cheiro, uma sensação etc. Do ponto de vista da teoria de revisão de crenças, interessa apenas o efeito desta entrada no estado epistêmico do agente.

Um problema próximo, porém diferente, do problema da revisão é o problema da *atualização de crenças* [2] [KM92]. Um agente atualiza suas crenças para se adaptar a uma mudança que ocorreu no mundo externo a ele. Por exemplo, um agente que acredita que um livro está em cima da estante deve atualizar suas crenças quando o livro é alterado de posição. Em contrapartida, um agente revisa suas crenças quando muda de idéia sobre algo. O mundo externo ao agente não mudou, o agente que mudou a sua forma de entender o mundo. Por exemplo, quando o agente revisa suas crenças e passa a acreditar que a manteiga é saudável não significa que a manteiga passou a ser saudável, mas que o agente passou a acreditar nisso. O disparador externo apenas faz o agente perceber isso. No primeiro caso o agente passa a acreditar que anteriormente o livro estava em cima da estante e que agora não está mais. No segundo caso o agente passa a acreditar que a manteiga sempre fora saudável e que anteriormente ele estava equivocado em acreditar o contrário.

Uma entrada epistêmica pode levar a diferentes tipos de mudanças no estado epistêmico. Tanto em conjunto de crenças como em bases de crenças vamos considerar três tipos de *mudanças de crenças*, ou *operações*, induzidas por alguma entrada epistêmica:

expansão: leva o agente a aceitar uma nova proposição.

revisão: leva o agente a aceitar uma proposição de forma consistente.

contração: leva um agente a abandonar uma proposição (torná-la indeterminada).

Resumindo, um agente está em um certo momento em um estado epistêmico específico. Então algum disparador externo faz com que ele mude de idéia e passe a estar em um segundo estado epistêmico. Uma mudança epistêmica seria a função que leva o primeiro estado epistêmico ao segundo estado epistêmico.

Um dos principais desafios da área de revisão de crenças é definir *critérios de racionalidade* para essas mudanças de crenças. O que esperar de um agente racional quando ele muda suas crenças. Muitos autores como Harman [Har86] apontam que qualquer agente racional deve, por exemplo, obedecer a algum *critério da minimalidade* ao mudar suas crenças. Ao executar uma mudança de crenças esperamos que o agente altere minimamente seu estado epistêmico, ou seja, esperamos que ele não abandone crenças desnecessariamente.

Nas seções 3.2.2 e 3.2.3 apresentaremos uma série de postulados de racionalidade para contração e para revisão em conjuntos de crenças. Nas seções 3.3.1 e 3.3.2 apresentaremos postulados para revisão em bases de crenças.

Para cada operação, além de definir os critérios, ou postulados de racionalidade, devemos mostrar como construí-la. Uma construção é *caracterizada* por um conjunto de postulados se, por um lado, ela satisfaz todos os postulados e, por outro, qualquer operação que satisfaça esses postulados pode ser obtida através dessa construção. O resultado que prova a equivalência entre uma construção e um conjunto de postulados é chamado de *teorema de representação*. Assim, para cada operação devemos apresentar: um conjunto de postulados de racionalidade, uma ou mais construções e teoremas de representação relacionando cada construção a um conjunto de postulados de racionalidade.

Um sistema dotado de um modelo para estados epistêmicos, uma definição de atitudes epistêmicas para cada elemento desse modelo, uma classe de entradas epistêmicas e um conjunto de operações de mudanças epistêmicas é chamado de *sistema de crenças*. Nesse capítulo apresentaremos dois sistemas de crenças: teoria AGM e teoria das bases de crenças.

Como fizemos no Capítulo 2 ao definirmos a LPC, podemos extrair dessa seção um roteiro para usarmos sempre que definirmos um novo sistema de crenças:

[2] Do inglês belief updates

1. definir como representar os estados epistêmicos,

2. enumerar as possíveis atitudes epistêmicas,

3. definir as possíveis operações:

   (a) definir os postulados de racionalidade,

   (b) definir uma construção,

   (c) provar o teorema de representação.

3.2 Teoria AGM

A teoria AGM [AGM85, Gär88] estuda mudanças de crenças em estados epistêmicos modelados como conjuntos logicamente fechados de sentenças.

Assumimos que o agente raciocína de acordo com uma lógica $\langle \mathscr{L}, Cn \rangle$ que satisfaz determinadas propriedades (voltaremos a esse ponto na seção 3.2.1). Essa lógica é chamada de *lógica subjacente* ao problema de revisão de crenças.

O estado epistêmico de um agente em um determinado momento é representado por um conjunto de sentenças logicamente fechado chamado *conjunto de crenças*. Ou seja, um conjunto de crenças é simplesmente um conjunto $K \in 2^{\mathscr{L}}$ tal que $K = Cn(K)$.

São considerados três tipos de atitudes epistêmicas perante uma sentença $\alpha \in \mathscr{L}$. Seja K o conjunto de crenças do agente, uma sentença α pode ser:

aceita: se $\alpha \in K$

rejeitada: se α é inconsistente com K

indeterminada: se $\alpha \notin K$ e α é consistente com K.

Três diferentes tipos de mudanças, ou operações, em conjuntos de crenças são admitidas: expansão, contração e revisão. Seja K o conjunto de crenças que representa o estado epistêmico de um agente, representaremos por $K + \alpha$, $K * \alpha$ e $K - \alpha$ os conjuntos de crenças desse agente após respectivamente expandí-lo, revisá-lo e contraí-lo por α.

A *expansão* é sem dúvida a mais simples dessas três operações. Uma expansão consiste na aceitação simples de uma sentença $\alpha \in \mathscr{L}$ em um estado epistêmico representado por K. Gärdenfors [Gär88] define um conjunto de postulados para expansão (omitidos aqui) e conclui que esses postulados são equivalentes à seguinte construção: $K + \alpha = Cn(K \cup \{\alpha\})$.

A *revisão* é a aceitação consistente de uma sentença. Além de garantir que $\alpha \in K * \alpha$ e que $K * \alpha$ é consistente temos que garantir algum critério de minimalidade nesta mudança. Note que a revisão é bem mais complicada do que a expansão, pois a revisão envolve componentes "extra-lógicos".

Considere o exemplo 1.1: ao conhecer a dieta do Dr. Atkins o agente deve remover algumas de suas crenças anteriores para aceitar de forma consistente as crenças que manteiga e margarina são saudáveis, porém, nenhum critério totalmente lógico é capaz de decidir qual sentença deve ser removida.

Diferente da expansão, não definiremos o operador $*$ explicitamente. Ao invés disso, restringimos as possibilidades de $*$ através de postulados de racionalidade apresentados na seção 3.2.3.

A *contração* , por sua vez, consiste na remoção de uma sentença $\alpha \in \mathscr{L}$ de um conjunto de crenças K. Além de garantir que $\alpha \notin K - \alpha$, devemos garantir algum critério de minimalidade nessa mudança. Como na revisão, esta operação depende de fatores "extra-lógicos". Postulados para contração serão apresentados na seção 3.2.2.

3.2.1 Suposições AGM

Na teoria AGM o estado de crenças de um agente é modelado como um conjunto logicamente fechado de sentenças. Portanto, assumimos que existe uma lógica subjacente ao problema de revisão de crenças. Definimos essa lógica como um par $\langle \mathscr{L}, Cn \rangle$ em que $\mathscr{L}$ é a linguagem, ou seja o conjunto de sentenças da lógica, e Cn é o operador de consequência.

Grande parte da literatura de revisão de crenças assume que essa lógica satisfaz certas propriedades chamadas *suposições AGM*. Assume-se que $\mathscr{L}$ é fechada por todos conectivos lógicos convencionais ($\wedge$, $\vee$, $\rightarrow$ e $\neg$) e que $\langle \mathscr{L}, Cn \rangle$ satisfaz:

tarskianicidade: a lógica é monotônica, idempotente e satisfaz inclusão (veja a seção 2.3).

compacidade: se $\alpha \in Cn(K)$ então existe $K' \subseteq K$ finito tal que $\alpha \in Cn(K')$

dedução: $\alpha \in Cn(K)$ sse $\beta \rightarrow \alpha \in Cn(K)$

supra-classicalidade: toda consequência da lógica proposicional clássica é consequência de $\langle \mathscr{L}, Cn \rangle$.

Um exemplo de lógica que satisfaz essas suposições é a LPC apresentada no Capítulo 2. Uma consequência das suposições AGM é a propriedade chamada explosão inconsistente:

explosão inconsistente: Se K é inconsistente então para qualquer $\beta \in \mathscr{L}$ temos que $\beta \in Cn(K)$.

Essa propriedade mostra que existe um único conjunto de crenças inconsistente, o conjunto $\mathscr{L}$ de todas as sentenças da linguagem.

A maior contribuição do presente trabalho é explorar as consequências de abandonar parte dessas propriedades. Nesse capítulo mostraremos a abordagem tradicional de revisão de crenças em que essas propriedades são assumidas.

3.2.2 Contração AGM

Nessa seção definiremos a operação de contração em conjuntos de crenças. Apresentaremos os postulados de racionalidade chamados *postulados AGM para contração* [AGM85]. Em seguida apresentaremos uma construção para a contração chamada *partial meet* ou *intersecção parcial*. Por fim, enunciaremos o teorema que prova a equivalência entre a contração partial meet e os postulados AGM: o teorema de representação para partial meet.

Postulados AGM

Quando um agente busca questionar a veracidade de alguma de suas crenças, em uma argumentação por exemplo, ele executa uma contração:

> **Exemplo 3.1:** Acredito que manteiga não é saudável $\neg s$ e que portanto não devo comer muita manteiga $\neg s \rightarrow \neg c$. Ao conversar com um nutricionista que acredita que devo comer manteiga, posso querer contrair $\neg c$ para o bem da argumentação.

Uma contração modifica o estado epistêmico de um agente. Ou seja, é uma operação que leva um estado epistêmico a um novo estado epistêmico.

A contração AGM é uma operação $-$ em um conjunto de crenças K que leva uma sentença α a se tornar indeterminada. Ou seja, a operação $-$ procura remover α de K. A sentença α é chamada *entrada* pois representa uma entrada epistêmica. O resultado dessa operação é um novo conjunto de crenças representado simbolicamente por $K - \alpha$.

Formalmente, temos uma lógica $\langle \mathscr{L}, Cn \rangle$ e um conjunto $K \in 2^{\mathscr{L}}$ tal que $K = Cn(K)$. A operação $-$ é definida para K e leva entradas $\alpha \in \mathscr{L}$ em subconjuntos de $\mathscr{L}$.

Como argumentamos anteriormente, uma contração exige componentes extra-lógicos. Logo, não definiremos uma função $-$ diretamente. Ao invés disso apresentaremos postulados que restringem as possibilidades de $-$.

O primeiro desses postulados é o fecho. Como estamos representando estados epistêmicos por conjuntos de crenças, nosso primeiro postulado deve garantir que $K - \alpha$ seja um conjunto de crenças:

(fecho) $K - \alpha = Cn(K - \alpha)$

Exemplo 3.2: O postulado do fecho garante que o resultado da contração é um conjunto de crenças. Representamos na figura 3.1 todos os conjuntos de crenças distintos na lógica proposicional clássica cuja assinatura é $\mathbb{P} = \{s, c\}$. Cada nó do diagrama da figura 3.1 representa um conjunto de crenças e as setas representam a relação de inclusão entre os conjuntos. No Capítulo 5 formalizaremos essa representação. Por hora a usaremos apenas para ilustrar os exemplos.

O fecho garante que o resultado da contração de $K = Cn(\{\neg s, \neg s \to \neg c\}) = Cn(\neg(s \vee c))$ por $\neg c$ é um dos nós desse diagrama.

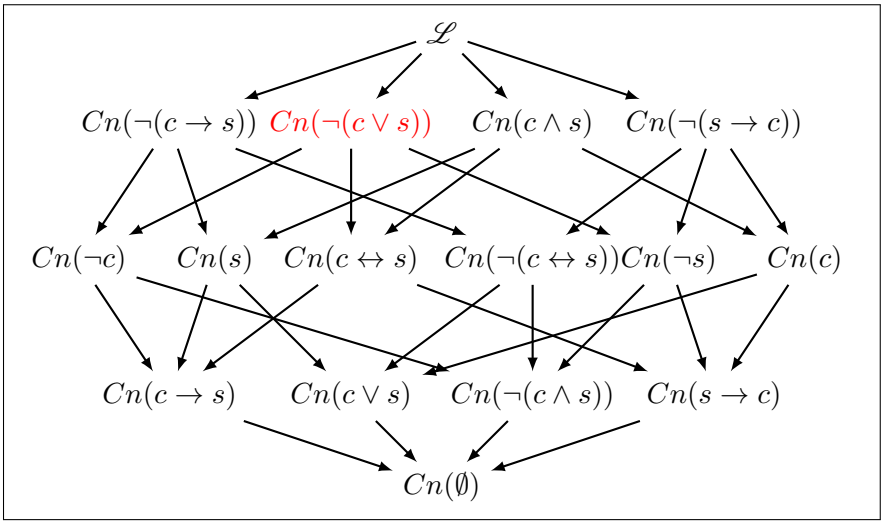


Figura 3.1: Diagrama do exemplo 3.2.

Note que o menor conjunto que satisfaz o fecho é $Cn(\emptyset)$ e que o maior é o conjunto de crenças inconsistente $\mathscr{L}$.

A contração é a modificação do estado epistêmico de um agente que leva uma sentença a não ser mais aceita. Ou seja, uma contração bem sucedida deve satisfazer $\alpha \notin K - \alpha$. Porém, se α é uma tautologia ($\alpha \in Cn(\emptyset)$) teríamos que violar o fecho para garantir $\alpha \notin K - \alpha$. Portanto, o próximo postulado é definido como:

(sucesso) Se $\alpha \notin Cn(\emptyset)$ então $\alpha \notin Cn(K - \alpha)$

Uma mudança epistêmica deveria levar em conta apenas o conteúdo das sentenças envolvidas e não sua forma. Assim, se partimos de um conjunto de crenças K e o contraímos por α deveríamos obter o mesmo resultado ao partir de K e contrair uma sentença β equivalente a α:

(extensionalidade) Se $Cn(\alpha) = Cn(\beta)$ então $K - \alpha = K - \beta$

Exemplo 3.3: Em nosso diagrama, o sucesso junto da extensionalidade garantem que o resultado da contração não pode ser um conjunto de crenças que contenha $Cn(\neg c)$. A figura 3.2 destaca os possíveis resultados de $K - \neg c$ em que $-$ satisfaz fecho, sucesso e extensionalidade.

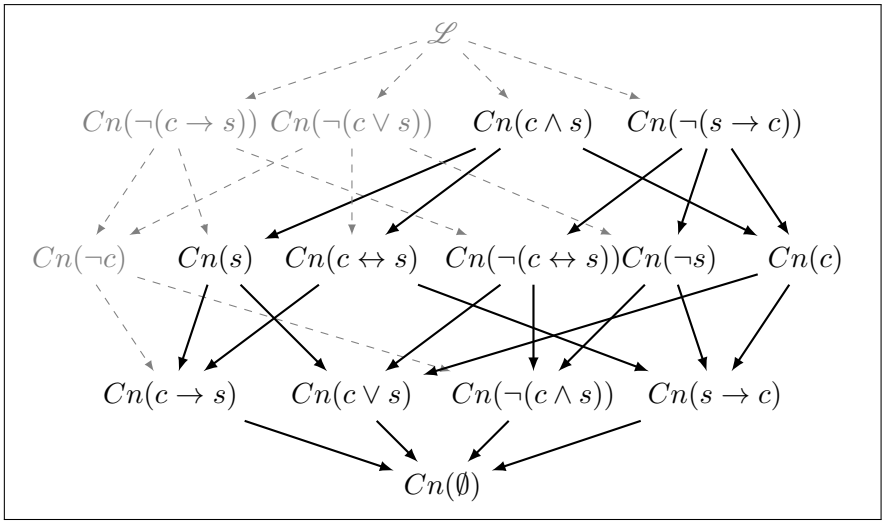


Figura 3.2: Diagrama do exemplo 3.3.

Os próximos postulados procuram garantir formas de minimalidade. Para remover uma sentença α de um conjunto de crenças K não é necessário adicionar novas sentenças. Assim, como buscamos uma mudança mínima temos:

(inclusão) $K - \alpha \subseteq K$

Exemplo 3.4: A inclusão garante que o conjunto de crenças $K - \neg c$ deve estar contido em K. Ou seja, $K - \neg c$ deve estar abaixo de K no diagrama da figura 3.2. O diagrama 3.3 mostra os possíveis $K - \neg c$ em que $-$ satisfaz fecho, sucesso, extensionalidade e inclusão.

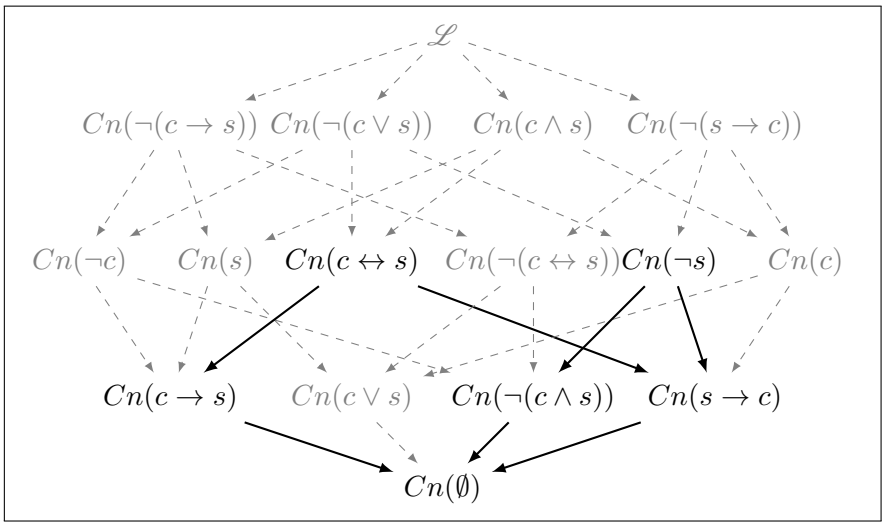


Figura 3.3: Diagrama do exemplo 3.4.

Um caso extremo da operação de contração ocorre quando α não é aceita pelo agente mesmo antes da contração ($\alpha \notin K$). Nesse caso, não precisamos modificar o conjunto de crenças do agente para que ele deixe de aceitar α. Assim, como buscamos modificar minimamente o estado de crenças do agente temos:

(vacuidade) Se $\alpha \notin K$ então $K - \alpha = K$

Uma operação que satisfaz todos os postulados apresentados até aqui é chamada de "withdrawal" [Mak87]. Hansson [Han91] argumenta que essa operação não garante a minimalidade de forma satisfatória. Considere a seguinte função $-$ para K:

$$K - \alpha = \begin{cases} K & \text{se } \alpha \notin Cn(K) \\ Cn(\emptyset) & \text{caso contrário.} \end{cases}$$

Essa operação nada intuitiva satisfaz todos os postulados da operação "withdrawal" para qualquer α. Nesse exemplo, caso o agente acreditasse em α, ao contrair α ele abandona todas suas crenças não tautológicas. Isso mostra que a operação "withdrawal" não garante a minimalidade de forma satisfatória.

O último e mais controverso postulado procura amenizar esse problema. A *recuperação* garante que se contraímos uma sentença α de K e em seguida expandimos o novo conjunto de crenças por α, devemos recuperar o conjunto de crenças original:

(recuperação) $(K - \alpha) + \alpha = K$

Esse postulado garante que a contração seja pequena o suficiente de forma que a re-adição de α em $K - \alpha$ recupere todo conjunto K. A recuperação é um importante critério de minimalidade, porém, bastante polêmica. Diversos autores criticam esse postulado [Mak87, Han91, Nie91, Lev91]. Por enquanto aceitaremos o postulado da recuperação, pois, apesar das críticas, ele é amplamente usado na literatura. Voltaremos a discutir a recuperação no Capítulo 6.

Exemplo 3.5: No exemplo que estamos explorando temos que a recuperação garante que $\neg(s \vee c) - \neg c, \neg c \vDash \neg(s \vee c)$. Ou seja, pela dedução temos que a recuperação garante que $\neg(s \vee c) - \neg c \vDash \neg c \to \neg(s \vee c)$. Equivalentemente temos que $\neg(s \vee c) - \neg c \vDash s \to c$. Ou seja, os possíveis $K - \neg c$ para um operador $-$ que satisfaz a recuperação devem estar acima de $s \to c$ no diagrama da figura 3.3. A figura 3.4 destaca essas possibilidades.

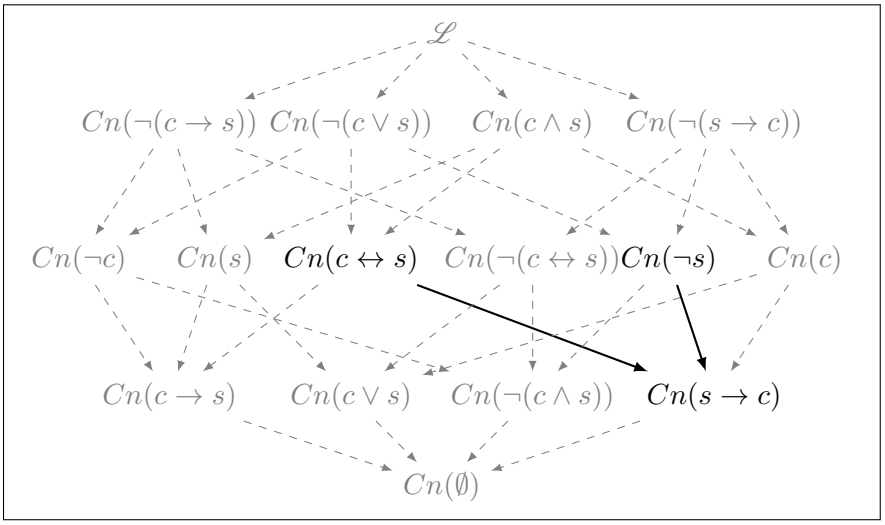


Figura 3.4: Diagrama do exemplo 3.5.

Os seis postulados apresentados até aqui são chamados *postulados AGM básicos para contração* [AGM85]. São eles:

(fecho) $K - \alpha = Cn(K - \alpha)$

(sucesso) Se $\alpha \notin Cn(\emptyset)$ então $\alpha \notin K - \alpha$

(inclusão) $K - \alpha \subseteq K$

(vacuidade) Se $\alpha \notin K$ então $K - \alpha = K$

(recuperação) $(K - \alpha) + \alpha = K$

(extensionalidade) Se $Cn(\alpha) = Cn(\beta)$ então $K - \alpha = K - \beta$

Em [AGM85] os autores apresentam mais dois postulados chamados de auxiliares. Não trataremos desses postulados em nossa apresentação. No restante desse trabalho nos referiremos aos postulados AGM básicos para contração simplesmente por postulados AGM para contração.

Postulados AGM Generalizados

Em LPC todo conjunto finito $\{a_1, \ldots, a_n\}$ é equivalente a uma sentença $a_1 \wedge \cdots \wedge a_n$, ou seja, $Cn(\{a_1, \ldots, a_n\}) = Cn(a_1 \wedge \cdots \wedge a_n)$. Nos próximos capítulos apresentaremos diversas lógicas não clássicas e em particular lógicas de descrição. Lógicas de descrição não são fechadas por $\wedge$ e, portanto, nem sempre é possível representar conjuntos finitos por um única sentença. Portanto, convém generalizar a operação $-$ de forma a aceitar um conjunto de sentenças como entrada. Tal operação é chamada na literatura de contração múltipla [FH94].

Não estamos interessados nas nuances dos resultados sobre conjuntos infinitos. Para facilitar nossa apresentação vamos definir $-$ como uma operação em um conjunto de crenças arbitrário K cuja entrada é um conjunto finitamente representável A.

Definição 3.6 (conjunto finitamente representável)**.** *Um conjunto A é* finitamente representável *se existe A' finito tal que $Cn(A) = Cn(A')$.*

A seguinte generalização dos postulados AGM para contração foi proposta por Fuhrmann e Hansson em [FH94] e posteriormente defendida por Flouris em [Flo06, FPA06]:

(fecho) $K - A = Cn(K - A)$

(sucesso) Se $A \notin Cn(\emptyset)$ então $A \notin K - A$

(inclusão) $K - A \subseteq K$

(vacuidade) Se $A \notin K$ então $K - A = K$

(recuperação) $(K - A) + A = K$

(extensionalidade) Se $Cn(A) = Cn(B)$ então $K - A = K - B$

Esta generalização é bastante direta, mas não é a única encontrada na literatura. A contração múltipla apresentada acima interpreta um conjunto de sentenças como uma conjunção. Assim para $K - A$ deve remover pelo menos um elemento de A (não necessariamente todos). Fuhrmann [Fuh97, FH94] chama esse tipo de contração de contração-escolha em contraposição a contração-pacote que remove todos elementos de A. Em nossa abordagem vamos considerar apenas a contração-escolha como apresentada acima.

Contração Partial Meet

Nessa seção apresentaremos uma construção para a contração AGM.

A literatura de revisão de crenças apresenta diversas construções para contração que satisfazem os postulados AGM [AGM85, Gro88, Gär88, Rot92]. Apresentaremos aqui apenas a contração *partial meet* ou *intersecção parcial* [AGM85].

Uma forma de garantir a minimalidade da operação de contração é considerar os maiores subconjuntos do conjunto de crenças K que não implicam α. O conjunto de todos esses subconjuntos de K é chamado de *conjunto resíduo*[3]:

Definição 3.7 (conjunto resíduo [AGM85])**.** *Seja K uma conjunto de crenças e seja α uma sentença. O* conjunto resíduo *$K \perp \alpha$ é o conjunto cujos elementos X são subconjuntos maximais de K que não implicam α. Ou seja, $X \in K \perp \alpha$ sse:*

[3] Do inglês remainder set.

1. $X \subseteq K$ *(subconjunto de* K*)*

2. $\alpha \notin Cn(X)$ *(que não implica* α*)*

3. *se* $X \subset X' \subseteq K$ *então* $\alpha \in Cn(X')$ *(maximal)*

Exemplo 3.8: A Figura 3.1 representa todos conjuntos de crenças na LPC com assinatura $\mathbb{P} = \{c, s\}$. Considerando o conjunto de crenças $K = Cn(\{\neg s, \neg s \rightarrow \neg c) = Cn(\neg(s \vee c))$ temos que $K \perp \neg c = \{Cn(c \leftrightarrow s), Cn(\neg s)\}$.

Considere uma função γ que seleciona alguns elementos de $K \perp \alpha$ sempre que possível e devolve K caso contrário. Intuitivamente γ seleciona aqueles conjuntos que contém as crenças que o agente acredita de forma mais arraigada[4]. Formalmente:

Definição 3.9 (função de seleção [AGM85])**.** *Seja* K *um conjunto de crenças. Uma* função de seleção *para* K *é uma função* γ *tal que para todo* α*:*

1. $\emptyset \neq \gamma(K \perp \alpha) \subseteq K \perp \alpha$ *se* $K \perp \alpha \neq \emptyset$*.*

2. $\gamma(K \perp \alpha) = K$ *caso contrário*

A contração partial meet é a intersecção dos subconjuntos maximais de K que não implicam α que o agente acredita de forma mais arraigada. Ou seja, a contração partial meet é a intersecção dos conjuntos escolhidos pela função de seleção.

Formalmente, seja γ uma função de seleção para K, definimos a operação $-_\gamma$ como:

$$K -_\gamma \alpha = \bigcap \gamma(K \perp \alpha)$$

Exemplo 3.10: Seja $K = Cn(\{c, s\})$ um conjunto de crenças e considere que queremos contrair $\alpha = \neg c$. Temos então que:

$$K \perp \alpha = \{Cn(c \leftrightarrow s), Cn(\neg s)\}$$

Poderíamos ter então $\gamma(K \perp \alpha) = K \perp \alpha$, ou seja, a função de seleção poderia escolher ambos os conjuntos. Nesse caso:

$$K -_\gamma \alpha = Cn(c \leftrightarrow s) \cap Cn(\neg s) = Cn(s \rightarrow c)$$

Uma contração $-$ para K é uma *contração partial meet* sse existe função de seleção γ para K tal que para todo α temos que $K - \alpha = K -_\gamma \alpha$.

Os postulados AGM *caracterizam precisamente* a contração partial meet. Ou seja, qualquer contração partial meet satisfaz os postulados AGM e se uma operação $-$ satisfaz os postulados AGM para K então existe alguma função de seleção γ para K tal que $K - \alpha = K -_\gamma \alpha$ para qualquer sentença α.

Esse resultado mostra uma equivalência entre as funções que podem ser construídas pelo método partial meet e as funções admitidas pelos postulados AGM. Essa equivalência é chamada de *teorema de representação*.

Teorema de Representação 3.11. *[AGM85] Uma operação* $-$ *é uma contração partial meet sse* $-$ *satisfaz fecho, sucesso, inclusão, vacuidade, recuperação e extensionalidade.*

[4]Do inglês *epistemically entrenched*.

3.2.3 Revisão AGM

A revisão AGM é uma operação $*$ em um conjunto de crenças K que adiciona consistentemente uma nova sentença α. O resultado dessa operação deve ser um novo conjunto $K * \alpha$. Formalmente temos que $*$ é uma função em K que dada uma sentença devolve um conjunto de sentenças.

Como no caso da contração AGM, não definiremos a função $*$ diretamente, mas apresentaremos postulados de racionalidade para revisão. Apresentamos também uma construção e o teorema de representação que relaciona os postulados à construção.

A revisão consiste na adição consistente de uma nova sentença α ao conjunto de crenças K. Assim temos os seguintes postulados:

(sucesso) $\alpha \in K * \alpha$

(consistência) Se α é consistente então $K * \alpha$ é consistente

Alguns postulados de racionalidade para revisão são análogos a postulados para contração. Nesses casos seguiremos a tradição da literatura e usaremos os mesmos nomes. Sempre que houver risco de ambiguidade explicitaremos se nos referimos ao postulado da revisão ou ao postulado da contração.

(fecho) $K * \alpha = Cn(K * \alpha)$

(inclusão) $K * \alpha \subseteq K + \alpha$

(extensionalidade) Se $Cn(\alpha) = Cn(\beta)$ então $K * \alpha = K * \beta$

(vacuidade) Se $K + \alpha$ é consistente então $K * \alpha = K + \alpha$

O seguinte postulado foi apresentado no trabalho original [AGM85], mas não tem sido usado nos trabalhos mais recentes na literatura de revisão de crenças:

(identidade de Harper) $K * \neg\alpha \cap K = K - \alpha$ para alguma operação $-$ para K que satisfaz os postulados para contração

Makinson [Mak87] argumenta que esse postulado não deveria ser usado como um postulado propriamente dito, mas como uma forma de construir uma contração a partir de uma revisão.

Assim, os postulados AGM para revisão como definidos em [Gär88] são:

(fecho) $K * \alpha = Cn(K * \alpha)$

(sucesso) $\alpha \in K * \alpha$

(inclusão) $K * \alpha \subseteq K + \alpha$

(vacuidade) Se $K + \alpha$ é consistente então $K * \alpha = K + \alpha$

(consistência) Se α é consistente então $K * \alpha$ é consistente

(extensionalidade) Se $Cn(\alpha) = Cn(\beta)$ então $K * \alpha = K * \beta$

Pelos mesmos argumentos apresentados na seção anterior, convém definir uma generalização da revisão para o caso em que a entrada seja um conjunto de sentenças. A generalização que apresentaremos é bastante natural e também foi apresentada em [Flo06]. Como no caso da contração assumimos que K é um conjunto de crenças e que a entrada A é um conjunto finitamente representável.

(fecho) $K * A = Cn(K * A)$

(sucesso) $A \subseteq K * A$

(inclusão) $K * A \subseteq K + A$

(vacuidade) Se $K + A$ é consistente então $K * A = K + A$

(consistência) Se A é consistente então $K * A$ é consistente

(extensionalidade) Se $Cn(A) = Cn(B)$ então $K * A = K * B$

Identidade de Levi

A forma mais usada para definir uma construção para a revisão AGM é a partir de uma contração $-$ através da identidade de Levi. Suponha que $-$ é uma contração para K que satisfaz todos os postulados AGM. Uma revisão $*$ para K definida pela identidade de Levi [AM82] é construída como:

$$K * \alpha = K - \neg\alpha + \alpha$$

Podemos, então, usar a contração partial meet para definir uma construção para a revisão. Uma revisão definida pela identidade de Levi usando uma contração partial meet é chamada de *revisão partial meet*.

Qualquer revisão partial meet satisfaz todos os postulados AGM para revisão [AGM85]. Além disso, como no caso da contração, os postulados AGM para revisão caracterizam precisamente a revisão partial meet. Ou seja, o seguinte teorema de representação vale para a revisão partial meet:

Teorema de Representação 3.12. *[Han99, Gär88, AGM85] Seja K um conjunto de crenças. Uma operação $*$ para K é uma revisão partial meet sse $*$ satisfaz fecho, sucesso, inclusão, vacuidade, consistência e extensionalidade.*

A identidade de Levi mostra como definir uma revisão a partir de uma contração. Como argumentamos anteriormente, é possível usar a identidade de Harper para definir uma contração a partir de uma revisão. Os resultados a seguir mostram como essas duas construções estão relacionadas:

Teorema 3.13. *[Mak87] Seja $-$ uma contração para K que satisfaz os postulados AGM e seja $*$ a revisão para K obtida pela identidade de Levi a partir de $-$. Para todo sentença α temos que $K \cap K * \neg\alpha = K - \alpha$.*

Teorema 3.14. *[Mak87] Seja $*$ uma revisão para K que satisfaz os postulados AGM e seja $-$ a contração para K obtida pela identidade de Harper a partir de $*$. Para todo sentença α temos que $K - \neg\alpha + \alpha = K * \alpha$.*

3.3 Bases de Crenças

A *teoria das bases de crenças* estuda a dinâmica dos estados epistêmicos representados como conjuntos arbitrários de sentenças B. Esse sistema de crenças admite quatro atitudes epistêmicas em relação a uma sentença α:

rejeitada: se α não é consistente com B

aceita explicitamente: se $\alpha \in B$

aceita implicitamente: se $\alpha \in Cn(B) \setminus B$

indeterminada: se $\alpha \notin Cn(B)$ e α é consistente com B.

Diremos simplesmente que uma sentença α é *aceita* se ela for aceita explicita ou implicitamente, ou seja, se $\alpha \in Cn(B)$.

A teoria de bases de crenças, diferente da teoria AGM, distingue aquilo que o agente acredita explicitamente daquilo que ele acredita meramente como consequência de suas crenças explicitas.

Alguns autores como Nebel [Neb90a] usam o termo base de crenças para se referir a uma representação finita de um conjunto de crenças. Nossa abordagem não segue esses autores. Seguimos a abordagem de Fuhrmann, Hansson e Wassermann [Fuh91, Han99, Was00b] que consideram bases de crenças como um sistema de crenças distinto do sistema baseado em conjuntos de crenças que diferencia crenças explicitas de crenças implícitas.

O exemplo 1.2 ilustra bem a diferença entre bases de crenças e conjuntos de crenças. Em bases de crenças, determinadas crenças implícitas são automaticamente abandonadas quando abandonamos uma crença explícita. A esse processo damos o nome de *propagação de descrença* [MS88].

Como na teoria AGM, as três principais mudanças de crenças são a expansão, a revisão e a contração.

A *expansão* é a mudança que leva uma sentença α a ser aceita por um agente cujo estado epistêmico é representado pelo conjunto de sentenças B. Como em bases de crenças o estado epistêmico do agente é representado simplesmente por um conjunto de sentenças, a forma mais simples de definir a expansão é $B + \alpha = B \cup \{\alpha\}$.

Para a *contração* e para a *revisão*, como na teoria AGM, apresentaremos postulados de racionalidade. Mostraremos construções para cada uma dessas operações e provaremos os respectivos teoremas de representação.

3.3.1 Contração

A contração em bases de crenças é a mudança de crenças que leva uma sentença α a se tornar indeterminada. Ou seja, se inicialmente o agente aceita a sentença α (explicita ou implicitamente), depois da contração o agente não deve mais ter opinião sobre α. Além disso, pelo princípio da minimalidade, desejamos que a mudança seja mínima.

A maioria dos postulados AGM deve valer também para bases de crenças, i.e. conjuntos não fechados. Assim temos:

(sucesso) Se $\alpha \notin Cn(\emptyset)$ então $\alpha \notin Cn(B - \alpha)$

(extensionalidade) Se $Cn(\alpha) = Cn(\beta)$ então $B - \alpha = B - \beta$

(inclusão) $B - \alpha \subseteq B$

(vacuidade) Se $\alpha \notin B$ então $B - \alpha = B$

Em bases de crenças o postulado da inclusão é bem mais controverso do que em conjuntos de crenças.

> **Exemplo 3.15:** Considere o exemplo 3.1. A base de crenças que representa o estado epistêmico inicial do agente é $B = \{\neg s, \neg s \rightarrow \neg c\}$. Pela inclusão temos que o resultado da contração entre B e $\neg c$ tem de ser um dos seguintes conjuntos: $\{\neg s, \neg s \rightarrow \neg c\}$, $\{\neg s\}$, $\{\neg s \rightarrow \neg c\}$ ou $\emptyset$. O primeiro deles viola o sucesso, pois implica $\neg c$.

O fecho evidentemente não é satisfeito em bases de crenças. Já a recuperação em bases de crenças não é compatível com a inclusão e com o sucesso. Considere o seguinte exemplo:

> **Exemplo 3.16:** [Han99] Seja $-$ uma operação em uma base $B = \{p \wedge q\}$ que satisfaz sucesso e inclusão então $B - p \subseteq \emptyset$. Porém $\emptyset \cup \{p\} = \{p\} \neq B$.

Isso se dá porque o postulado da inclusão em bases de crenças exige muito mais do que em conjuntos de crenças. No apêndice A voltaremos a tratar da inclusão em bases de crenças e de sua relação com a recuperação. Por hora precisamos de algum postulado alternativo que garanta alguma tipo de minimalidade em bases de crenças. Na literatura são encontrados dois postulados para esse fim sugeridos por Hansson [Han91]:

(relevância) Se $\beta \in B$ e $\beta \notin B - \alpha$ então existe B' tal que $B - \alpha \subseteq B' \subseteq B$ e $\alpha \notin Cn(B')$, mas $\alpha \in Cn(B' \cup \{\beta\})$

(core-retainment) Se $\beta \in B$ e $\beta \notin B - \alpha$ então existe B' tal que $B' \subseteq B$ e $\alpha \notin Cn(B')$, mas $\alpha \in Cn(B' \cup \{\beta\})$

A idéia desses postulados é garantir que só sejam removidos de B aquelas sentenças β que de alguma forma ajudam a derivar α.

Na presença de core-retainment temos que o postulado da vacuidade é redundante. Ou seja:

Proposição 3.17. *[Han99] Seja $-$ uma operação para B. Se $-$ satisfaz core-retainment então $-$ satisfaz a vacuidade.*

O postulado da extensionalidade procura garantir que sentenças equivalentes sejam contraídas de um mesmo conjunto de crenças de forma equivalente. Em bases de crenças esse postulado é um pouco fraco. Vamos usar em seu lugar o seguinte postulado:

(uniformidade) Se é verdade que para todo subconjunto B' de B vale que $\alpha \in Cn(B')$ sse $\beta \in Cn(B')$ então $B - \alpha = B - \beta$

Temos dois conjuntos de postulados que diferem no critério de minimalidade. Cada um deles corresponde a uma construção que apresentaremos nas próximas seções: contração kernel e contração partial meet. O primeiro conjunto de postulados contém a *relevância* como principal critério de minimalidade enquanto o outro contém *core-retainment*. Ambos contém, além do critério de minimalidade, o *sucesso*, a *inclusão* e a *uniformidade*.

Como argumentamos nas seções anteriores, convém sempre generalizar os postulados para receber conjuntos de sentenças como entrada. A generalização da inclusão e do sucesso já foram apresentadas. A generalização dos demais postulados é bastante direta:

(relevância) Se $\beta \in B$ e $\beta \notin B - A$ então existe B' tal que $B - A \subseteq B' \subseteq B$ e $A \nsubseteq Cn(B')$, mas $A \subseteq Cn(B' \cup \{\beta\})$

(core-retainment) Se $\beta \in B$ e $\beta \notin B - A$ então existe B' tal que $B' \subseteq B$ e $A \nsubseteq Cn(B')$, mas $A \subseteq Cn(B' \cup \{\beta\})$

(uniformidade) Se é verdade que para todo subconjunto B' de B vale que $A_1 \subseteq Cn(B')$ sse $A_2 \subseteq Cn(B')$ então $B - A_1 = B - A_2$.

Contração Partial Meet

Como na teoria AGM, a contração partial meet $B -_\gamma \alpha$ é definida a partir do conjunto resíduo $B \perp \alpha$ e de uma função de seleção γ como:

$$B -_\gamma \alpha = \bigcap \gamma(B \perp \alpha)$$

B não é um conjunto de crenças, mas uma base de crenças.

Exemplo 3.18: Seja $B = \{\neg s, \neg s \rightarrow \neg c\}$. O conjunto resíduo $B \perp \neg c$ é formado pelos subconjuntos de B maximais que não implicam $\neg c$. Os subconjuntos de B são $\{\neg s, \neg s \rightarrow \neg c\}$, $\{\neg s\}$, $\{\neg s \rightarrow \neg c\}$ e $\emptyset$. O primeiro implica $\neg c$ e o último está incluso nos demais (não é minimal). Portanto

$$B \perp \neg c = \{\{\neg s\}, \{\neg s \rightarrow \neg c\}\}$$

Uma função de seleção válida seria $\gamma(B \perp \neg c) = \{\{\neg s\}\}$. Nesse caso:

$$B -_{\gamma} \neg c = \{\neg s\}$$

Como adiantamos, essa construção é totalmente caracterizada pelos postulados *sucesso*, *inclusão*, *uniformidade* e *relevância*. Formalmente temos:

Teorema de Representação 3.19 ([HW02]). *Seja $\langle \mathscr{L}, Cn \rangle$ uma lógica monotônica e compacta. A operação $-$ é uma contração partial meet para B sse $-$ satisfaz:* sucesso, inclusão, uniformidade *e* relevância.

Note que esse teorema é aplicável a uma classe maior de lógicas (qualquer lógica compacta e monotônica) do que os teoremas de representação AGM que assumem que a lógica subjacente satisfaz as suposições AGM.

Contração Kernel

Na contração partial meet consideramos os maiores subconjuntos de B que não implicam α. A *contração kernel* [Han94], ao invés disso, considera os menores subconjuntos de B que implicam α. Formalmente temos:

Definição 3.20 (kernel [Han94]). *Seja B uma base de crenças e seja α uma sentença. O* conjunto kernel $B \perp\!\!\!\perp \alpha$ *é o conjunto tal que $X \in B \perp\!\!\!\perp \alpha$ sse:*

1. *$X \subseteq B$ (subconjunto de K)*
2. *$\alpha \in Cn(B)$ (que implica α)*
3. *se $X' \subset X$ então $\alpha \notin Cn(X')$ (minimal)*

Intuitivamente o conjunto kernel contém todas as formas mínimas de derivar α a partir de B.

> **Exemplo 3.21:** Seja $B = \{\neg s, \neg s \rightarrow \neg c\}$. O conjunto kernel $B \perp\!\!\!\perp \neg c$ é formado pelos subconjuntos de B minimais que implicam $\neg c$. Nenhum subconjunto próprio de B implica $\neg c$. Logo, $B \perp\!\!\!\perp \neg c = \{B\}$.

A contração kernel consiste em quebrar cada uma dessas formas de derivar α e, assim, impedir que α seja inferido. Ou seja, na contração kernel removemos pelo menos uma sentença de cada elemento do kernel. Para tanto, é preciso definir uma função que escolha os elementos que serão removidos. Essa função é chamada de função de incisão:

Definição 3.22 (função de incisão [Han94]). *Seja B uma base de crenças. Uma* função de incisão *para B é uma função σ tal que para todo α:*

1. *$\sigma(B \perp\!\!\!\perp \alpha) \subseteq \bigcup B \perp\!\!\!\perp \alpha$ e*
2. *Se $\emptyset \neq X \in B \perp\!\!\!\perp \alpha$ então $X \cap \sigma(B \perp\!\!\!\perp \alpha) \neq \emptyset$*

Intuitivamente a função de incisão escolhe as sentenças que o agente acredita de forma menos arraigada.

Seja σ uma função de incisão para B. Definimos a operação $-_{\sigma}$ como:

$$B -_{\sigma} \alpha = B \setminus \sigma(B \perp\!\!\!\perp \alpha)$$

Exemplo 3.23: Seja $B = \{p_1, p_2, p_1 \vee p_2 \rightarrow q\}$ uma base de crenças. Suponha que desejamos contrair $\alpha = q$. Temos então que:

$$\begin{aligned} B \perp\!\!\!\perp \alpha \quad = \quad & \{ \\ & \{p_1, p_1 \vee p_2 \rightarrow q\}, \\ & \{p_2, p_1 \vee p_2 \rightarrow q\}\} \end{aligned}$$

Uma função de incisão deve escolher pelo menos uma sentença de cada elemento de $B \perp\!\!\!\perp \alpha$. Um exemplo de função de incisão válida seria $\sigma = \{p_1, p_2\}$. Nesse caso:

$$B -_\sigma \alpha = \{p_1 \vee p_2 \rightarrow q\}$$

A contração kernel é completamente caracterizada pelos postulados *sucesso*, *inclusão*, *uniformidade* e *core-retainment*. Ou seja:

Teorema de Representação 3.24 ([HW02])**.** *Seja $\langle \mathscr{L}, Cn \rangle$ uma lógica monotônica e compacta. A operação $-$ é uma contração kernel para B sse $-$ satisfaz:* sucesso*,* inclusão*,* uniformidade *e* core-retainment*.*

3.3.2 Revisão

A revisão em bases de crenças é a operação $*$ que leva uma sentença a ser aceita explicitamente pelo agente de forma consistente. Na teoria AGM mostramos que é possível definir uma operação de revisão a partir de uma contração usando a identidade de Levi [AM82]:

$$B - \neg\alpha + \alpha$$

No caso de bases de crenças também é possível definir a revisão revertendo a identidade de Levi da seguinte forma [Han93]:

$$B + \alpha - \neg\alpha$$

Note que não é possível usar essa segunda construção em conjuntos de crenças, pois se $K + \alpha$ é inconsistente então, pela explosão inconsistente, temos que $K + \alpha = \mathscr{L}$. Logo, ao expandir K por α perdemos toda informação que tínhamos sobre o conjunto de crenças original K.

Isso não acontece em bases de crenças, pois podemos ter bases inconsistentes distintas. Por exemplo:

$$\begin{aligned} B_1 &= \{p \wedge \neg p\} \\ B_2 &= \{q, p, \neg p\} \end{aligned}$$

A revisão em bases de crenças construída pela identidade de Levi é chamada de *revisão interna* enquanto que a revisão construída pela identidade de Levi reversa é chamada de *revisão externa*. Ambas as revisões satisfazem os seguintes postulados:

(sucesso) $\alpha \in B * \alpha$

(inclusão) $B * \alpha \subseteq B + \alpha$

(consistência) Se α é consistente então $B * \alpha$ é consistente

Além desses postulados ambas as revisões devem satisfazer algum critério de minimalidade, core-retainment ou relevância, dependendo se a contração usada para gerar a revisão é uma contração kernel ou uma contração partial meet.

(core-retainment) Se $\beta \in B \setminus B * \alpha$ então existe $B' \subseteq B \cup \{\alpha\}$ tal que $B' \cup \{\alpha\}$ é consistente, mas $B' \cup \{\alpha, \beta\}$ não é.

(relevância) Se $\beta \in B \setminus B * \alpha$ então existe B' tal que $B * \alpha \subseteq B' \subseteq B \cup \{\alpha\}$ e $B' \cup \{\alpha\}$ é consistente, mas $B' \cup \{\alpha, \beta\}$ não é.

A revisão interna satisfaz, além desses postulados, a uniformidade:

(uniformidade) Se vale que para todo $B' \subseteq B$, $B' \cup \{\alpha\}$ é consistente sse $B' \cup \{\beta\}$ é consistente então $B \cap (B * \alpha) = B \cap (B * \beta)$.

A revisão externa satisfaz os três primeiros postulados, uma versão fraca de uniformidade e um postulado chamado de pré-expansão:

(uniformidade fraca) Se $\alpha, \beta \in B$ e vale que para todo $B' \subseteq B$, $B' \cup \{\alpha\}$ é consistente sse $B' \cup \{\beta\}$ é consistente então $B \cap (B * \alpha) = B \cap (B * \beta)$

(pré-expansão) $B + \alpha * \alpha = B * \alpha$

Temos, então, quatro diferentes revisões para bases de crenças:

revisão interna kernel: $B *_\sigma \alpha = (B \setminus \sigma(B \perp\!\!\!\perp \neg\alpha)) + \alpha$

revisão externa kernel: $B *_\sigma \alpha = (B + \alpha) \setminus \sigma((B + \alpha) \perp\!\!\!\perp \neg\alpha)$

revisão interna partial meet: $B *_\gamma \alpha = (\bigcap \gamma(B \perp \neg\alpha)) + \alpha$

revisão externa partial meet: $B *_\gamma \alpha = \bigcap \gamma((B + \alpha) \perp \neg\alpha)$

Como no caso da contração em bases de crenças, não precisamos de todas as suposições AGM para provar os teoremas de representação para revisão em bases de crenças. Além da monotonicidade e da compacidade, para provar os teoremas de representação para revisão em bases de crenças assumimos que a lógica $\langle \mathscr{L}, Cn \rangle$ é fechada por negação e que essa negação satisfaz a seguinte propriedade introduzida em [HW02]:

não contravenção local α: Se $\neg\alpha \in Cn(B \cup \{\alpha\})$ então $\neg\alpha \in Cn(B)$

Esta propriedade garante que a negação de α é sempre irrelevante para provar α.

Estamos agora em condição de enunciar os teoremas de representação para revisões em bases de crenças. Todos eles assumem que a lógica subjacente seja compacta, monotônica e satisfaça a não contravenção local α:

Teorema de Representação 3.25 ([HW02])**.** *A operação $*$ é uma revisão interna kernel para B sse $*$ satisfaz:* sucesso*,* inclusão*,* uniformidade *e* core-retainment*.*

Teorema de Representação 3.26 ([HW02])**.** *A operação $*$ é uma revisão externa kernel para B sse $*$ satisfaz:* sucesso*,* inclusão*,* uniformidade fraca*,* pré-expansão *e* core-retainment*.*

Teorema de Representação 3.27 ([HW02])**.** *A operação $*$ é uma revisão interna partial meet para B sse $*$ satisfaz:* sucesso*,* inclusão*,* uniformidade *e* relevância*.*

Teorema de Representação 3.28 ([HW02])**.** *A operação $*$ é uma revisão externa partial meet para B sse $*$ satisfaz:* sucesso*,* inclusão*,* uniformidade fraca*,* pré-expansão *e* relevância*.*

3.3.3 Semi-Revisão

A operação de revisão que apresentamos na seção anterior assume que o agente deve aceitar a nova sentença α. Hansson em [Han97] estuda uma operação parecida com a revisão chamada de *semi-revisão*. Hansson argumenta que a revisão só pode ser aplicada depois que o agente tenha decidido aceitar α. A semi-revisão delega a tarefa de aceitar α para o mecanismo de seleção (função de seleção ou função de incisão). Dessa forma, a semi-revisão não satisfaz o postulado sucesso.

Construímos uma semi-revisão adicionando α à base B e em seguida removendo as inconsistências. Durante o processo de remover as inconsistências (*consolidação*) a sentença α pode vir a ser removida dependendo do mecanismo de seleção. Definiremos duas formas para remover inconsistências.

A primeira se baseia no conjunto kernel. Seja σ uma função de incisão. A semi-revisão kernel $?_\sigma$ para B é definida como:

$$B?_\sigma\alpha = (B + \alpha) \setminus \sigma((B + \alpha) \perp\!\!\!\perp \bot)$$

Com relação aos postulados, a semi-revisão kernel difere da revisão externa kernel apenas pela ausência do sucesso e pela substituição da uniformidade fraca pelo seguinte postulado:

(troca interna) Se $\alpha, \beta \in B$ então $B?\alpha = B?\beta$

Assim, a semi-revisão kernel é caracterizada por *consistência*, *inclusão*, *core-retainment*, *pré-expansão* e *troca interna*.

Teorema de Representação 3.29. *[Was00b] Seja $\langle \mathscr{L}, Cn\rangle$ uma lógica compacta e monotônica. Um operador $?$ é um operador de semi-revisão kernel para B sse para todo α o operador $?$ satisfaz:* consistência*,* inclusão*,* core-retainment*,* pré-expansão *e* troca interna*.*

A segunda forma de semi-revisão é a semi-revisão partial meet. Essa operação $?_\gamma$ para B é definida como:

$$B?_\gamma\alpha = \bigcap \gamma((B + \alpha) \perp \bot)$$

A semi-revisão partial meet é caracterizada por *consistência*, *inclusão*, *relevância*, *pré-expansão* e *troca interna*.

Teorema de Representação 3.30. *[Was00b] Seja $\langle \mathscr{L}, Cn\rangle$ uma lógica compacta e monotônica. Um operador $?$ é um operador de semi-revisão partial meet para B sse para todo α o operador $?$ satisfaz:* consistência*,* inclusão*,* relevância*,* pré-expansão *e* troca interna*.*

Neste capítulo tratamos da teoria AGM em lógicas que satisfazem as suposições AGM. No decorrer deste trabalho apresentaremos diversas lógicas que não satisfazem algumas dessas suposições como as lógicas de descrição que serão introduzidas no próximo capítulo.

Capítulo 4

Lógicas de Descrição

Lógicas de Descrição (LDs) [BCM$^+$03] são formalismos usados para representar (modelar) relações entre conceitos e indivíduos em um domínio. Em LDs podemos, por exemplo, expressar que o conceito "alimento saudável" é equivalente ao conceito "alimento não rico em gordura" ($Saudável \equiv Alimento \sqcap \neg \exists ricoEm.Gordura$) ou que manteiga é uma instância do conceito "alimento rico em gordura" (($Alimento \sqcap \exists ricoEm.Gordura)(manteiga)$) e dessas sentenças podemos inferir que manteiga não é uma instância de "alimento saudável" (($Alimento \sqcap \neg Saudável)(manteiga)$).

Diferente de seus predecessores, sistemas de frames [Min81] e redes semânticas [Qui67], as lógicas de descrição possuem semântica bem definida. Além disso, diferente da lógica de primeira ordem, elas são decidíveis (possuem método de decisão para os problemas de inferência).

Ao modelar um domínio, por um lado, é desejável um formalismo expressivo, capaz de representar uma vasta gama de problemas. Por outro lado, é importante garantir que os problemas de inferência (e.g.verificar se uma base de conhecimento possui um modelo) sejam decidíveis, e que de preferência possam ser resolvidos de forma eficiente. Como argumentado em [LB87], muitas vezes esses dois objetivos não são compatíveis. Um dos grandes desafios do estudo das lógicas de descrição é a busca por esse balanço entre expressividade e complexidade computacional.

O estudo das lógicas de descrição culminou com a adoção da linguagem OWL como linguagem padrão para representação de ontologias na web. A última versão da linguagem OWL (2.0) é uma variação sintática de uma lógica de descrição bastante expressiva ($\mathcal{SROIQ}$).

Nas próximas seções apresentaremos a linguagem e a semântica das lógicas de descrição (seção 4.1). Como existem diversas lógicas de descrição com diferentes linguagens e diferentes operadores de conseqüência lógica, usaremos uma lógica particular como exemplo: a lógica $\mathcal{ALC}$. Na seção 4.2 apresentaremos extensões da lógicas $\mathcal{ALC}$. Apresentaremos na seção 4.3 a linguagem OWL e, por fim, na seção 4.4 mostraremos a relação das lógicas de descrição com lógicas modais e com a lógica de primeira ordem.

4.1 Linguagem, Semântica e Tableaux

Nesta seção definiremos uma lógica de descrição chamada $\mathcal{ALC}$. Usaremos a definição de $\mathcal{ALC}$ para definir outras LDs nas seções posteriores. Seguindo o roteiro do fim do Capítulo 2:

1. definiremos a linguagem de $\mathcal{ALC}$.
2. definiremos a classe dos modelos de $\mathcal{ALC}$.
3. mostraremos como verificar se um modelo satisfaz uma sentença.
4. definiremos o operador de conseqüência a partir dos modelos.
5. mostraremos um método de decisão para o problema da satisfatibilidade baseado em tableaux.
6. mostraremos propriedades satisfeitas pelas LDs no Capítulo 5

4.1.1 Linguagem

As linguagens das lógicas de descrição são divididas em duas partes. A primeira, chamada de *linguagem de descrição*, define os conceitos da lógica. A segunda define as sentenças propriamente ditas. Cada LD possui sua própria linguagem.

A *assinatura* de uma linguagem de descrição é formada por três conjuntos disjuntos entre si $\langle N_C, N_R, N_I \rangle$. N_C é o conjunto de conceitos atômicos, N_R é o conjunto de papéis atômicos e N_I é o conjunto de indivíduos. Considere a linguagem de descrição da lógica $\mathcal{ALC}$ em que $A \in N_C$ e $R \in N_R$.[1]

$$C := A \mid \top \mid \bot \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \forall R.C \mid \exists R.C$$

Os símbolos $\neg$, $\sqcap$, $\sqcup$, $\forall$ e $\exists$ são chamados *construtores*. A linguagem de descrição de uma lógica é caracterizada pelos construtores que ela admite. A lógica $\mathcal{ALC}$ por exemplo, admite negação ($\neg$), conjunção ($\sqcap$), disjunção ($\sqcup$), restrição de valor ($\forall$) e a restrição existencial qualificada ($\exists$).

Exemplo 4.1: Considere a seguinte assinatura para a lógica $\mathcal{ALC}$: $\langle N_C, N_R, N_I \rangle$ em que:

$$\begin{aligned} N_C &= \{\textit{Alimento, Gordura, Trans, Saudável, Açúcar}\} \\ N_R &= \{\textit{ricoEm}\} \\ N_I &= \emptyset \end{aligned}$$

Os seguintes conceitos pertencem à linguagem de descrição desta lógica:

$\textit{Gordura}$: conjunto das gorduras.

$\exists \textit{ricoEm}.\textit{Gordura}$: conjunto de tudo o que é rico em pelo menos uma gordura.

$\neg\exists \textit{ricoEm}.\textit{Gordura}$: conjunto de tudo o que não é rico em nenhuma gordura.

$\textit{Alimento} \sqcap \neg\exists \textit{ricoEm}.\textit{Gordura}$: conjunto dos alimentos que não são ricos em nenhuma gordura.

$\textit{Alimento} \sqcap \neg\exists \textit{ricoEm}.\textit{Trans}$: conjunto dos alimentos que não são ricos em nenhuma gordura trans.

$\textit{Alimento} \sqcap \exists \textit{ricoEm}.\textit{Açúcar}$: conjunto dos alimentos que são ricos em algum açúcar.

Uma vez definida a linguagem de descrição de uma LD, definimos sua linguagem $\mathscr{L}$. Tradicionalmente, a linguagem $\mathscr{L}$ é dividida em duas partes, o TBox e o ABox.

O TBox contém sentenças que representam o conhecimento conceitual, terminológico, da forma:

$$\alpha := C \sqsubseteq C \mid C \equiv C$$

Ou seja, o TBox possui sentenças como

$$\textit{Saudável} \equiv \textit{Alimento} \sqcap \neg\exists \textit{ricoEm}.\textit{Gordura}$$

que define o conceito de alimento saudável como aqueles alimentos que não são ricos em gordura.

O Abox contém sentenças que representam conhecimento assertivo[2] da forma:

$$\alpha := C(a) \mid R(a,b) \mid a = b \mid a \neq b$$

em que C é um conceito qualquer, $a, b \in N_I$ e $R \in N_R$. O ABox contém sentenças como $(\textit{Alimento} \sqcap \exists \textit{ricoEm}.\textit{Gordura})(\textit{manteiga})$ que afirma que a manteiga é uma instância do conceito de alimento rico em gordura.

[1] Usamos letras maiúsculas A, C e D para conceitos em lógicas de descrição e as letras R e S para papéis.

[2] Do inglês "assertional"

A linguagem $\mathscr{L}_{\mathcal{ALC}}$ contém todas as sentenças tanto do ABox quanto do TBox. Usaremos os símbolos $\mathscr{L}^{TBox}$ e $\mathscr{L}^{ABox}$ para nos referir aos conjuntos de sentenças na linguagem do TBox e do ABox respectivamente.

Uma *base de conhecimento* em lógica de descrição é um par $\Sigma = \langle \mathcal{T}, \mathcal{A} \rangle$ em que $\mathcal{T} \subseteq \mathscr{L}^{TBox}$ e $\mathcal{A} \subseteq \mathscr{L}^{ABox}$.

Exemplo 4.2: Considere a seguinte assinatura:

$$\begin{aligned} N_C &= \{Saudável, Gordura, Alimento, Trans\} \\ N_R &= \{ricoEm\} \\ N_I &= \{manteiga, margarina\} \end{aligned}$$

As seguintes sentenças pertencem a linguagem $\mathscr{L}^{TBox}_{\mathcal{ALC}}$ com esta assinatura:

$Saudável \equiv Alimento \sqcap \neg \exists ricoEm.Gordura$

$Saudável \sqsubseteq Alimento \sqcap \neg \exists ricoEm.Trans$

$Trans \sqsubseteq Gordura$

$\exists ricoEm.Trans \sqsubseteq \exists ricoEm.Gordura$

Já as seguintes sentenças pertencem a linguagem $\mathscr{L}^{ABox}_{\mathcal{ALC}}$ com a mesma assinatura:

$(Alimento \sqcap \exists ricoEm.Gordura)(manteiga)$

$(Alimento \sqcap \exists ricoEm.Trans)(margarina)$

$(Alimento \sqcap \neg Saudável)(manteiga)$

$(Alimento \sqcap \neg Saudável)(margarina)$

4.1.2 Semântica

Definiremos a semântica das lógicas de descrição usando modelos. A classe dos modelos para um LD é formada por *interpretações* da forma $\mathcal{I} = \langle \Delta^{\mathcal{I}}, .^{\mathcal{I}} \rangle$. $\Delta^{\mathcal{I}}$ é um conjunto não vazio chamado *domínio* e $.^{\mathcal{I}}$ é uma *função de interpretação*. A função de interpretação associa conceitos atômicos a subconjuntos do domínio ($C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ para $C \in N_C$), papéis atômicos a relações binárias no domínio ($R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ para $R \in N_R$) e indivíduos a elementos do domínio ($a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ para $a \in N_I$). Depois de interpretar os conceitos, papéis e indivíduos atômicos *estendemos* a interpretação aos demais conceitos de forma recursiva:

$$\begin{aligned} \top^{\mathcal{I}} &= \Delta^{\mathcal{I}} \\ \bot^{\mathcal{I}} &= \emptyset \\ (\neg C)^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\ (C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\ (C \sqcup D)^{\mathcal{I}} &= C^{\mathcal{I}} \cup D^{\mathcal{I}} \\ (\forall R.C)^{\mathcal{I}} &= \{a : \forall (a,b) \in R^{\mathcal{I}} \rightarrow b \in C^{\mathcal{I}}\} \\ (\exists R.C)^{\mathcal{I}} &= \{a : \exists (a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\} \end{aligned}$$

Exemplo 4.3: Considere uma interpretação $\mathcal{I} = \langle \Delta^{\mathcal{I}}, .^{\mathcal{I}} \rangle$ em que:

$$\begin{aligned}
\Delta^{\mathcal{I}} &= \{manteiga, chocolate, margarina, cenoura, t_1, g_1, a_1\} \\
Alimento^{\mathcal{I}} &= \{manteiga, chocolate, margarina, cenoura\} \\
Trans^{\mathcal{I}} &= \{t_1\} \\
Gordura^{\mathcal{I}} &= \{g_1, t_1\} \\
\text{Açúcar}^{\mathcal{I}} &= \{a_1\} \\
Saudável^{\mathcal{I}} &= \{cenoura\} \\
ricoEm^{\mathcal{I}} &= \{(manteiga, g_1), (margarina, t_1), (chocolate, a_1)\}
\end{aligned}$$

Podemos estender $\mathcal{I}$ a conceitos complexos como

$$Alimento \sqcap \neg \exists ricoEm.Gordura$$

aplicando recursivamente as regras acima:

$$\begin{aligned}
(Alimento \sqcap \neg \exists ricoEm.Gordura)^{\mathcal{I}} &= Alimento^{\mathcal{I}} \cap (\neg \exists ricoEm.Gordura)^{\mathcal{I}} \\
&= Alimento^{\mathcal{I}} \cap \\
&\quad (\Delta^{\mathcal{I}} \setminus (\exists ricoEm.Gordura)^{\mathcal{I}}) \\
&= Alimento^{\mathcal{I}} \cap \\
&\quad (\Delta \setminus \{a : \exists (a,b) \in ricoEm^{\mathcal{I}} \wedge b \in Gordura^{\mathcal{I}}\}) \\
&= Alimento^{\mathcal{I}} \cap \\
&\quad (\Delta \setminus \{manteiga, margarina\}) \\
&= Alimento^{\mathcal{I}} \cap \\
&\quad \{cenoura, chocolate, t_1, g_1, a_1\} \\
&= \{cenoura, chocolate\}
\end{aligned}$$

Uma interpretação $\mathcal{I}$ *satisfaz*, ou é modelo para, uma sentença α (escrevemos $\mathcal{I} \vDash \alpha$) de acordo com as seguintes condições:

$$\begin{aligned}
\mathcal{I} \vDash C \sqsubseteq D &\quad \text{sse} \quad C^{\mathcal{I}} \subseteq D^{\mathcal{I}} \\
\mathcal{I} \vDash C(a) &\quad \text{sse} \quad a^{\mathcal{I}} \in C^{\mathcal{I}} \\
\mathcal{I} \vDash R(a,b) &\quad \text{sse} \quad (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}} \\
\mathcal{I} \vDash a = b &\quad \text{sse} \quad a^{\mathcal{I}} = b^{\mathcal{I}} \\
\mathcal{I} \vDash a \neq b &\quad \text{sse} \quad a^{\mathcal{I}} \neq b^{\mathcal{I}}
\end{aligned}$$

O operador de consequência lógica é definido de forma usual. Ou seja, $\alpha \in Cn(\Sigma)$ sse para todo $\mathcal{I}$ se $\mathcal{I}$ satisfaz todas as sentenças de Σ (tanto as sentenças do TBox, quanto as do ABox) então $\mathcal{I}$ satisfaz α.

Exemplo 4.4: A sentença $\exists ricoEm.Trans \sqsubseteq \exists ricoEm.Gordura$ é consequência de $Trans \sqsubseteq Gordura$. Ou seja, $\exists ricoEm.Trans \sqsubseteq \exists ricoEm.Gordura \in Cn(Trans \sqsubseteq Gordura)$.

Para mostrar que a afirmação acima vale temos que mostrar que para todo $\mathcal{I}$, se $Trans^{\mathcal{I}} \subseteq Gordura^{\mathcal{I}}$ então $(\exists ricoEm.Trans)^{\mathcal{I}} \subseteq (\exists ricoEm.Gordura)^{\mathcal{I}}$. Considere $\mathcal{I}$ tal que $Trans^{\mathcal{I}} \subseteq Gordura^{\mathcal{I}}$ e $a \in (\exists ricoEm.Trans)^{\mathcal{I}}$, então $a \in \{b : \exists (b,b') \in ricoEm^{\mathcal{I}} \wedge b' \in Trans^{\mathcal{I}}\}$. Como $Trans^{\mathcal{I}} \subseteq Gordura^{\mathcal{I}}$, temos que $a \in \{b : \exists (b,b') \in ricoEm^{\mathcal{I}} \wedge b' \in Gordura^{\mathcal{I}}\}$. Concluímos que se $a \in (\exists ricoEm.Trans)^{\mathcal{I}}$ então $a \in (\exists ricoEm.Gordura)^{\mathcal{I}}$ e, portanto, $(\exists ricoEm.Trans)^{\mathcal{I}} \subseteq (\exists ricoEm.Gordura)^{\mathcal{I}}$

Usamos lógicas de descrição para modelar um domínio. Por exemplo, o domínio dos alimentos saudáveis e não saudáveis. Escolhemos conceitos relacionados a esse domínio: alimento saudável, gordura etc. Esses conceitos representam conjuntos de elementos do domínio. O TBox restringe as possíveis interpretações desses conceitos. Por exemplo, se a sentença $Trans \sqsubseteq Gordura$ ocorre no TBox não aceitamos qualquer interpretação para $Trans$, mas somente aquelas em que $Trans$ é um subconjunto de $Gordura$. O mesmo vale para a interpretação dos indivíduos e o ABox. O ABox restringe as interpretações dos indivíduos.

4.1.3 Tableaux

Apresentaremos o método de prova para $\mathcal{ALC}$ baseado em tableaux [HNSS90] em três etapas. Primeiro mostramos um método para decidir se uma sentença do TBox é válida. Ou seja, se $\alpha \in Cn(\langle \emptyset, \emptyset \rangle)$ para $\alpha \in \mathscr{L}_{\mathcal{ALC}}^{TBox}$. Depois mostraremos como adaptar esse algoritmo para verificar se uma sentença qualquer é consequência de um ABox ($\alpha \in Cn(\langle \emptyset, \mathcal{A} \rangle)$). Por fim, mostramos como verificar se uma sentença qualquer é consequência de uma base de conhecimento (ABox e TBox).

Antes de definir o método do tableaux, note que $C \equiv D$ é equivalente a $\{C \sqsubseteq D, D \sqsubseteq C\}$ e que $C \sqsubseteq D$ é válido sse para $a \in N_I$ qualquer, $(C \sqcap \neg D)(a)$ é insatisfatível. Ou seja, o problema da validade de uma sentença do TBox pode ser reduzido ao problema da satisfatibilidade de um ABox.

Para facilitar a apresentação assumiremos que todos os conceitos estão na forma normal negada (FNN). Ou seja, que o símbolo $\neg$ ocorre apenas na frente de conceitos atômicos. Podemos passar qualquer conceito para sua forma normal negada "empurrando para dentro" as negações usando as seguintes equivalências:

$$\begin{aligned} \neg(A \sqcap B) &\equiv \neg A \sqcup \neg B \\ \neg(A \sqcup B) &\equiv \neg A \sqcap \neg B \\ \neg(\forall R.A) &\equiv \exists R.(\neg A) \\ \neg(\exists R.A) &\equiv \forall R.(\neg A) \\ \neg\neg A &\equiv A \end{aligned}$$

Vamos apresentar um método de tableaux para decidir se um ABox cujos conceitos estão na forma normal negada (FNN) é satisfatível. O método que apresentaremos é uma adaptação do método apresentado em [BS01].

Os *ramos* nesse tableau são ABoxes (conjuntos $B \in 2^{\mathscr{L}^{ABox}}$). Dizemos que um ramo possui um conflito se ele contém $A(a)$ e $\neg A(a)$ para algum $A \in N_C$ e algum $a \in N_I$. Definimos ramo completo, ramo fechado e tableau fechado como no Capítulo 2. Ou seja, o ramo é completo se não há mais regras aplicáveis e é fechado se possui um conflito. Um tableau é fechado sse todos os ramos são fechados.

Começamos o processo com um único ramo $\mathcal{A}$ e aplicamos as regras do algoritmo 4.1 até que todos os ramos ou fechem ou fiquem completos. Ao fim desse processo o algoritmo devolve `insatisfatível` se o tableau for fechado e `satisfatível` caso contrário.

As principais fontes de complexidade desse tableau são as regras $\rightarrow \sqcup$ e $\rightarrow \exists$ [DLNN97]. A regra $\rightarrow \sqcup$ não é determinística, pois ela gera um novo ramo. A regra $\rightarrow \exists$ gera um novo indivíduo no ramo atual.

Exemplo 4.5: Suponha que queremos verificar se

$$\forall R.(A \sqcap B) \sqsubseteq \forall R.A \sqcap \forall R.B \in Cn(\emptyset)$$

Para mostrar essa inclusão temos que mostrar que o seguinte ABox é insatisfatível:

$$\forall R.(A \sqcap B) \sqcap \neg(\forall R.A \sqcap \forall R.B)(x)$$

Algoritimo 4.1 Tableaux para $\mathcal{ALC}$

regra $\rightarrow \sqcap$**:**

condição: $\mathcal{A}$ contém um indivíduo $C_1 \sqcap C_2(x)$, mas não contém ambos $C_1(x)$ e $C_2(x)$.
ação: $\mathcal{A} := \mathcal{A} \cup \{C_1(x), C_2(x)\}$.

regra $\rightarrow \sqcup$**:**

condição: $\mathcal{A}$ contém $C_1 \sqcup C_2(x)$, mas não contém $C_1(x)$ nem $C_2(x)$.
ação: cria o ABox $\mathcal{A}' := \mathcal{A} \cup \{C_1(x)\}$ e $\mathcal{A} := \mathcal{A} \cup \{C_2(x)\}$

regra $\rightarrow \exists$**:**

condição: $\mathcal{A}$ contém $\exists R.C(x)$, e não existe y tal que $A(y)$ e $R(x, y) \in \mathcal{A}$.
ação: $\mathcal{A} := \mathcal{A} \cup \{C(y), R(x, y)\}$ em que y é um novo indivíduo.

regra $\rightarrow \forall$**:**

condição: $\mathcal{A}$ contém $\forall R.C(x)$ e $R(x, y)$, e não contém $C(y)$
ação: $\mathcal{A} := \mathcal{A} \cup \{C(y)\}$.

Antes de aplicar o algoritmo temos de passar o conceito para sua forma normal negada:

$$\begin{aligned} \forall R.(A \sqcap B) \sqcap \neg(\forall R.A \sqcap \forall R.B) &\equiv \forall R.(A \sqcap B) \sqcap \\ &\quad (\forall \neg R.A \sqcup \neg \forall R.B) \\ &\equiv \forall R.(A \sqcap B) \sqcap \\ &\quad (\exists R.(\neg A) \sqcup \exists R.(\neg B)) \end{aligned}$$

Agora aplicamos os passos do algoritmo acima. Nos passos marcados com $*$ ocorreram conflitos:

$$\begin{array}{lcll} \mathcal{A}_0 & := & \{(\forall R.(A \sqcap B) \sqcap (\exists R.(\neg A) \sqcup \exists R.(\neg B)))(x)\} & \\ \mathcal{A}_0 & := & \mathcal{A}_0 \cup \{(\forall R.(A \sqcap B))(x), & \\ & & (\exists R.(\neg A) \sqcup \exists R.(\neg B))(x)\} & \rightarrow \sqcap \\ \mathcal{A}_1 & := & \mathcal{A}_0 \cup \{(\exists R.(\neg A))(x)\} & \rightarrow \sqcup \\ \mathcal{A}_1 & := & \mathcal{A}_1 \cup \{R(x, y), \underline{(\neg A)(y)}\} & \rightarrow \exists \\ \mathcal{A}_1 & := & \mathcal{A}_1 \cup \{(A \sqcap B)(y)\} & \rightarrow \forall \\ \mathcal{A}_1 & := & \mathcal{A}_1 \cup \{\underline{A(y)}, B(y)\} & \rightarrow \sqcap \quad * \\ \mathcal{A}_0 & := & \mathcal{A}_0 \cup \{(\exists R.(\neg B))(x)\} & \rightarrow \sqcup \\ \mathcal{A}_0 & := & \mathcal{A}_0 \cup \{R(x, y), \underline{(\neg B)(y)}\} & \rightarrow \exists \\ \mathcal{A}_0 & := & \mathcal{A}_0 \cup \{(A \sqcap B)(y)\} & \rightarrow \forall \\ \mathcal{A}_0 & := & \mathcal{A}_0 \cup \{A(y), \underline{B(y)}\} & \rightarrow \sqcap \quad * \end{array}$$

O tableau fechou, pois todos os ramos ($\mathcal{A}_0$ e $\mathcal{A}_1$) fecharam. Logo, o algoritmo deve retornar `insatisfatível`. Concluímos que para qualquer interpretação $\mathcal{I}$ temos que $(\forall R.(A \sqcap B))^{\mathcal{I}} \subseteq (\forall R.A \sqcap \forall R.B)^{\mathcal{I}}$

O procedimento que apresentamos é *correto*, pois, com exceção da regra $\rightarrow \sqcup$, todas as regras aplicadas a um ABox satisfatível gera um novo ABox satisfatível. A regra $\rightarrow \sqcup$ aplicada a um ABox satisfatível $\mathcal{A}$, por sua vez, gera um novo ramo $\mathcal{A}'$ em que $\mathcal{A}$ ou $\mathcal{A}'$ são satisfatíveis. Assim, o tableau só fecha se o ABox inicial for insatisfatível.

Como no tableau para LPC, provamos a *completude* mostrando um modelo canônico $\mathcal{I}_{\mathcal{A}}$ a partir de um ramo $\mathcal{A}$ completo:

- O domínio $\Delta^{\mathcal{I}_\mathcal{A}}$ é formado pelos indivíduos em $\mathcal{A}$.
- Para todo conceito B definimos $B^{\mathcal{I}_\mathcal{A}} = \{x | B(x) \in \mathcal{A}\}$.
- Para todo papel R definimos $R^{\mathcal{I}_\mathcal{A}} = \{(x, y) | R(x, y) \in \mathcal{A}\}$.

Por fim, o procedimento sempre *termina*, pois a ação de cada regra gera conceitos estritamente menores do que os conceitos considerados na condição.

Resumindo, em $\mathcal{ALC}$ o problema da insatisfatibilidade para um ABox, e conseqüentemente o problema da validade de uma sentença do TBox, é *decidível* (veja [BS01] para a demonstração completa). Mais precisamente, é possível provar que esse problema é PSPACE-completo [SSS91, Tob01].

O leitor familiarizado com lógicas modais irá notar que esse algoritmo se assemelha muito ao algoritmo para construir um tableau para a lógica multi-modal K_n. Como veremos em mais detalhes na seção 4.4, a lógica $\mathcal{ALC}$ é uma variante notacional da lógica K_n [Sch91].

Mostraremos agora como adaptar esse algoritmo para verificar se uma sentença qualquer é consequência de um ABox. Para facilitar a exposição vamos usar o símbolo $\neg\alpha$:

$$\neg\alpha = \begin{cases} (C \sqcap \neg D)(a) & \text{para um } a \text{ novo se} \quad \alpha = C \sqsubseteq D \\ \neg C(a) & \text{se} \quad \alpha = C(a) \\ \neg R(a, b) & \text{se} \quad \alpha = R(a, b) \\ a = b & \text{se} \quad \alpha = (a \neq b) \\ a \neq b & \text{se} \quad \alpha = (a = b) \end{cases}$$

Note que $\neg R(a, b)$ não faz parte de nossa linguagem, o usaremos como símbolo auxiliar para verificar se $R(a, b)$ é consequência de um ABox. A adição deste símbolo faz com que os ramos não sejam mais ABoxes, mas isso não compromete o algoritmo. Dizemos que um ramo possui um conflito se ele contém α e $\neg\alpha$.

Para verificar se uma sentença α é consequência de um ABox $\mathcal{A}$, assumindo que todos estão na FNN, usamos o método do tableau começando com o conjunto com um único ramo $\{\mathcal{A} \cup \{\neg\alpha\}\}$.

Verificar se uma sentença α é consequência de uma base de conhecimento com ABox e TBox é um pouco mais delicado. Vamos começar por um caso mais simples. Assumiremos primeiramente que o TBox possui as seguintes restrições:

- toda sentença $\alpha \in TBox$ é da forma $A \equiv C$ tal que $A \in N_C$. Chamamos sentenças dessa forma de *definições*.
- o $TBox$ não possui duas definições $A \equiv C$ e $A' \equiv C'$ tal que $A = A'$. Ou seja, o $TBox$ não possui *definições múltiplas*.
- o $TBox$ não contém sentenças da forma $A_1 \equiv C_1$, $A_n \equiv C_n$ tal que para todo i, A_i aparece em C_{i-1} e A_1 ocorre em C_n. Ou seja, o $TBox$ não possui *definições cíclicas*.

Um TBox que satisfaz as restrições acima é chamado de *taxonomia*. A ausência de definições múltiplas e de ciclos nos permite reduzir as tarefas de inferência em relação a uma taxonomia às tarefas de inferência em relação a taxonomia vazia. Para isso, basta substituir repetidamente cada ocorrência de um conceitos não primitivos, conceito que ocorre do lado esquerdo de alguma definição, por sua definição. Esse processo de substituição pode levar a uma explosão exponencial no tamanho da fórmula[3]. Porém, é possível mostrar que o problema de inferência em $\mathcal{ALC}$ em relação a uma taxonomia ainda é PSPACE-completo [HP06].

[3] A seguinte terminologia leva a uma explosão exponencial no tamanho da formula A_0 [Neb90b]: $A_0 \equiv \forall R.A_1 \sqcap \forall S.A_1, \ldots, A_{n-1} \equiv \forall R.A_n \sqcap \forall S.A_n$

Exemplo 4.6: O seguinte TBox é uma terminologia [BCM+03]:

$$\begin{aligned} Mulher &\equiv Pessoa \sqcap Fêmea \\ Homem &\equiv Pessoa \sqcap \neg Mulher \\ Mãe &\equiv Mulher \sqcap \exists temFilho.Pessoa \\ Pai &\equiv Homem \sqcap \exists temFilho.Pessoa \\ Avó &\equiv Mãe \sqcap \exists temFilho.(Pai \sqcup Mãe) \end{aligned}$$

Verificar se $Avó \sqsubseteq Mulher$ é consequência dessa terminologia é equivalente a verificar se seguinte sentença é válida:

$$\begin{aligned} &((Pessoa \sqcap Fêmea) \sqcap \exists temFilho.Pessoa) \sqcap \\ &\exists temFilho.((Pessoa \sqcap \neg Mulher) \sqcap \exists temFilho.Pessoa) \sqcup \\ &((Pessoa \sqcap Fêmea) \sqcap \exists temFilho.Pessoa)) \sqsubseteq Mulher \end{aligned}$$

Essa sentença é obtida da primeira por seguidas substituições:

$$\begin{aligned} Avó &\equiv Mãe \sqcap \exists temFilho.(Pai \sqcup Mãe) \\ &\equiv (Mulher \sqcap \exists temFilho.Pessoa) \sqcap \\ &\quad \exists temFilho.((Homem \sqcap \exists temFilho.Pessoa) \sqcup \\ &\quad (Mulher \sqcap \exists temFilho.Pessoa)) \\ &\equiv ((Pessoa \sqcap Fêmea) \sqcap \exists temFilho.Pessoa) \sqcap \\ &\quad \exists temFilho.((Pessoa \sqcap \neg Mulher) \sqcap \exists temFilho.Pessoa) \sqcup \\ &\quad ((Pessoa \sqcap Fêmea) \sqcap \exists temFilho.Pessoa)) \end{aligned}$$

Uma lógica admite *axiomas GCI* ("general concept inclusion") se seu TBox não possui as restrições acima. Em $\mathcal{ALC}$ qualquer conjunto de axiomas GCI ($\{C_1 \sqsubseteq D_1, \ldots, C_n \sqsubseteq D_n\}$) é equivalente a uma sentença da forma $\top \sqsubseteq C$ ($\top \sqsubseteq (\neg C_1 \sqcup D_1) \sqcap, \ldots, \sqcap (\neg C_n \sqcup D_n)$). Ou seja, podemos reduzir qualquer TBox a uma única sentença da forma $\top \sqsubseteq C$.

Para decidir se uma sentença α é consequência de uma base de conhecimento Σ qualquer, primeiro passamos o TBox para a forma $\top \sqsubseteq C$. Em seguida usamos o algoritmo baseado em tableaux apresentado nessa seção para verificar se α é consequência do ABox e a cada nova instância x criada em um ramo acrescentamos a sentença $C(x)$ neste ramo. Essa alteração, porém, quebra a garantia de término do algoritmo. Para recuperá-la devemos usar algum mecanismo que detecte ciclos no algoritmo [BS01].

Na presença de axiomas GCI, decidir se uma sentença é consequência de uma base de conhecimento é um problema ExpTime-completo. Ou seja, há um salto de complexidade quando permitimos o uso não restrito de sentenças no TBox. Voltaremos a esse ponto na seção 4.2 ao tratar de extensões da lógica $\mathcal{ALC}$.

4.1.4 Consistência

A literatura em lógicas de descrição apresenta diferentes definições de consistência. Nesta seção generalizaremos este conceito para abranger as diversas definições encontradas na literatura.

Como mostrado na seção 4.1.2, em LDs um *conceito* A é dito *satisfatível* se existe pelo menos uma interpretação que satisfaz aquele conceito, ou seja $A^{\mathcal{I}} \neq \emptyset$. Por exemplo, o conceito $A \sqcap \neg A$ não é satisfatível, pois $(A \sqcap \neg A)^{\mathcal{I}} = \emptyset$ para qualquer $\mathcal{I}$.

A presença de conceitos insatisfatíveis costuma indicar um erro de modelagem e normalmente é considerada indesejável. Porém, a presença de conceitos insatisfatíveis não compromete a integridade da base de conhecimentos como um todo, ainda é possível tirar conclusões úteis dessas bases de conhecimento.

Um outro cenário acontece quando a base de conhecimento que não possuem modelo algum. Nesse caso dizemos que a *base* é insatisfatível. O problema com essas bases de conhecimento é muito mais drástico do que com as primeiras, pois, neste caso nenhuma conclusão útil pode ser inferida da base.

Estas dois cenários são classificados em [FHP+06] da seguinte forma: bases do primeiro tipo são chamadas *incoerentes* e bases do segundo tipo de *inconsistentes*. Nesse trabalho não usaremos essa nomenclatura. No primeiro caso diremos que a base possui um elemento insatisfatível e no segundo que a própria base é insatisfatível.

O termos *inconsistente* será generalizado. Chamaremos de inconsistente qualquer base de conhecimento que possua consequências indesejáveis. O critério de consequência indesejável será definido a cada problema. Para cada problema assumiremos que existe um conjunto Ω de sentenças indesejáveis e uma base de conhecimento será dita inconsistente sse implicar algum elemento de Ω. Esta generalização é suficiente para englobar ambas situações apresentadas nesta seção. Voltaremos a este tópico no capítulo 8.

4.1.5 Modelagem

Para encerrar essa seção vamos modelar em $\mathcal{ALC}$ o exemplo apresentado na introdução dessa tese. Faremos isso como um exemplo simples de modelagem em lógica de descrição. Além disso, usaremos essa modelagem mais a frente como exemplo para revisão de crenças em lógicas de descrição.

No primeiro momento o agente do exemplo 1.1 acreditava que alimentos ricos em gordura não eram saudáveis e que tanto a manteiga quanto a margarina eram alimentos ricos em gordura. Podemos modelar essas crenças do agente da seguinte forma:

$$Saudável \sqsubseteq Alimento \sqcap \neg \exists rico_em.Gordura$$

$$Alimento \sqcap \exists rico_em.Gordura(margarina)$$

$$Alimento \sqcap \exists rico_em.Gordura(manteiga)$$

Dessas sentenças podemos inferir, por exemplo, que nem margarina nem manteiga são saudáveis ($\neg Saudável(margarina)$ e $\neg Saudável(manteiga)$).

No segundo momento o agente aprende que existe um tipo de gordura chamada trans, que a margarina é rica nesse tipo de gordura e que alimentos ricos nesse tipo de gordura não são saudáveis. Então o agente expande suas crenças pelo seguinte conjunto de sentenças:

$$Trans \sqsubseteq Gordura$$

$$Saudável \sqsubseteq Alimento \sqcap \neg \exists rico_em.Trans$$

$$\exists rico_em.Trans(margarina)$$

Por fim o agente revisa suas crenças ao conhecer a dieta do Dr. Atkins em que margarina e manteiga são considerados alimentos saudáveis. Modelamos a entrada dessa operação de revisão por:

$$Saudável(manteiga)$$

$$Saudável(margarina)$$

Essas duas sentenças junto das sentenças acima formam um conjunto de sentenças que não possui modelo em $\mathcal{ALC}$, ou seja, esse conjunto é insatisfatível.

4.2 Extensões de $\mathcal{ALC}$

Na seção anterior definimos a lógica $\mathcal{ALC}$, sua linguagem, sua semântica e mostramos um procedimento de prova baseado em tableaux. Nessa seção definiremos a linguagem e a semântica de extensões da lógica $\mathcal{ALC}$. Mostraremos como estender $\mathcal{ALC}$ com papéis transitivos, papéis inversos, hierarquia de papéis, nominais e restrições numéricas.

A lógica $\mathcal{S}$ admite, além dos construtores e tipos de axioma de $\mathcal{ALC}$, *papéis transitivos*. Assumimos que existe um subconjunto $N_{R_+} \subseteq N_R$ de papéis transitivos. Para toda interpretação $\mathcal{I}$ temos que se $R \in N_{R_+}$:

$$\text{se } (x,y) \in R^{\mathcal{I}} \text{ e } (y,z) \in R^{\mathcal{I}} \text{ então } (x,z) \in R^{\mathcal{I}}$$

Exemplo 4.7: Nas lógica $\mathcal{S}$ é possível expressar que a relação $ancestralDe$ é transitiva. Assim, se afirmo que meu avô é ancestral de meu pai ($ancestralDe(avo, pai)$) e meu pai é meu ancestral ($ancestralDe(pai, eu)$) então meu avô tem de ser meu ancestral ($ancestralDe(avo, eu)$).

Os nomes das demais extensões de $\mathcal{ALC}$ são formados acrescentando letras em $\mathcal{ALC}$ ou em $\mathcal{S}$. Assim, por exemplo, a lógica $\mathcal{SI}$ é a lógica que estende $\mathcal{ALC}$ com papéis transitivos e papéis inversos. O conjunto dos papéis de uma lógica que admite *papéis inversos* (ROL) é formado por $N_R \cup \{R^- : R \in N_R\}$. Para toda interpretação $\mathcal{I}$ temos que:

$$\text{se } (x,y) \in R^{\mathcal{I}} \text{ então } (y,x) \in (R^-)^{\mathcal{I}}$$

Claro que em uma lógica que admite papéis transitivos se $R \in N_{R_+}$ então R^- também é transitivo e deve ser interpretado como tal.

Para evitar usar símbolos como R^{--}, vamos definir uma abreviação:

$$Inv(R) = \begin{cases} R^- & se \quad R \in N_R \\ S & se \quad R = S^- \end{cases}$$

Exemplo 4.8: Na presença de papéis inversos podemos expressar que a relação $ancestralDe$ é inversa a relação $descendenteDe$ ou que a relação $irmaoDe$ é simétrica (inversa a si mesma). Assim se afirmamos que meu avô é meu ancestral ($ancestralDe(avo, eu)$) concluímos que eu sou descendente de meu avô ($descendenteDe(eu, avo)$).

As lógicas que admitem *hierarquia de papéis* possuem a letra $\mathcal{H}$ no nome. Essas lógicas possuem, além do TBox e do ABox, um RBox que contém sentenças da forma:

$$\alpha := R \sqsubseteq S$$

Seja $\mathcal{R}$ um RBox, definimos a relação $\boxed{\leq}$ como o fecho reflexivo-transitivo de $\sqsubseteq$ sobre $\mathcal{R} \cup \{Inv(R) \sqsubseteq Inv(S) : R \sqsubseteq S \in \mathcal{R}\}$ e chamamos de $\mathcal{R}^+$ o conjunto de axiomas $R \sqsubseteq S$ tal que $R \boxed{\leq} S$. Usaremos a abreviação $R \equiv S$ para indicar que $R \boxed{\leq} S$ e $S \boxed{\leq} R$.

Se $R \in N_{R_+}$ e $R \equiv S$ então S é transitivo. Usaremos o símbolo $Trans(R)$ para indicar que um papel é transitivo. Ou seja, temos $Trans(S)$ sse $R \equiv S$ para algum $S \in N_{R_+}$ ou $Inv(S) \in N_{R_+}$. Papéis não transitivos são chamados de *simples*.

Uma interpretação $\mathcal{I}$ satisfaz um RBox $\mathcal{R}$ (escrevemos $\mathcal{I} \vDash \mathcal{R}$) sse para toda sentença $R \sqsubseteq S \in \mathcal{R}$ temos que:

$$R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$$

Uma base de conhecimento em uma lógica que admite hierarquia de papéis é uma tripla $\langle \mathcal{T}, \mathcal{A}, \mathcal{R} \rangle$ em que $\mathcal{T}$ é um TBox, $\mathcal{A}$ é um ABox e $\mathcal{R}$ é um RBox. Uma interpretação $\mathcal{I}$ satisfaz uma base $\Sigma = \langle \mathcal{T}, \mathcal{A}, \mathcal{R} \rangle$ sse $\mathcal{I}$ satisfaz $\mathcal{T}$, $\mathcal{A}$ e $\mathcal{R}$. Uma sentença α é um consequência de Σ sse todo $\mathcal{I}$ que satisfaz Σ satisfaz α.

Exemplo 4.9: Em uma lógica que possui hierarquia de papéis podemos impor que as relações *paiDe* e *avoDe* estejam contidas na relação *ancestralDe*.

As lógicas que admitem *nominais* (representadas pela letra $\mathcal{O}$) são capazes de definir conceitos como uma enumeração de indivíduos $\{a_1, \ldots, a_n\}$ em que $a_1, \ldots, a_n \in N_I$. A interpretação desse conceito é simples:

$$\{a_1, \ldots, a_n\}^{\mathcal{I}} = \{a_1^{\mathcal{I}}, \ldots, a_n^{\mathcal{I}}\}$$

Em lógicas que admitem nominais qualquer sentença do ABox possui uma sentença equivalente do TBox:

$$\begin{aligned} C(a) &\equiv \{a\} \sqsubseteq C \\ R(a,b) &\equiv \{a\} \sqsubseteq \exists R.\{b\} \\ a = b &\equiv \{a\} \sqsubseteq \{b\} \\ a \neq b &\equiv \{a\} \sqsubseteq \neg\{b\} \end{aligned}$$

Exemplo 4.10: Na presença de nominais é possível representar o conjunto dos países da América do Norte como

$$\{EUA, Canadá, México\}$$

Uma LD que admite *restrição numérica qualificada* ($\mathcal{Q}$) é capaz de representar conceitos como $\leq_n R.C$ ($\geq_n R.C$) em que R é um papel simples[4] (não transitivo), n é um número inteiro positivo e C é um conceito qualquer. Os elementos de $\leq_n R.C$ ($\geq_n R.C$) são todos aqueles indivíduos que se relacionam com no máximo (pelo menos) n indivíduos de C através da relação R. Formalmente temos:

$$\begin{aligned} (\leq_n R.C)^{\mathcal{I}} &= \{b : \#\{(b,a) \in R^{\mathcal{I}} \wedge a \in C^{\mathcal{I}}\} \leq n\}) \\ (\geq_n R.C)^{\mathcal{I}} &= \{b : \#\{(b,a) \in R^{\mathcal{I}} \wedge a \in C^{\mathcal{I}}\} \geq n\}) \end{aligned}$$

Exemplo 4.11: Em lógicas que admitem restrições numéricas qualificadas podemos representar o conjuntos daqueles que possuem pelo menos dois filhos homens:

$$\geq_2 filho.Homem$$

Usamos a letra $\mathcal{N}$ para as lógicas que admitem restrição numérica simples. Ou seja, conceitos da forma $\leq_n R.\top$ e $\geq_n R.\top$. Se, além disso, restringirmos n a 1 dizemos que a lógica admite papéis funcionais ($\mathcal{F}$).

Em muitas situações concretas não queremos apenas raciocinar sobre classes abstratas como *Pessoa* ou *Animal*, mas também com tipos de dados como `string` ou `inteiro`. Para evitar a perda da decidibilidade [Lut02], ao tratar de tipos de dados devemos separar sua interpretação da interpretação dos demais papéis, indivíduos e conceitos. Para isso definimos um conjunto $\Delta_{\mathbf{D}}^{\mathcal{I}}$ disjunto de $\Delta^{\mathcal{I}}$ e definimos a interpretação de um *papel concreto* como um relação binária em $\Delta^{\mathcal{I}} \times \Delta_{\mathbf{D}}^{\mathcal{I}}$. Papéis concretos podem ser usados em restrições de valor, restrições existenciais e restrições numéricas. Acrescentemos a letra **D** entre parênteses ao fim do nome da lógica para indicar que a lógica admite papéis concretos.

A tabela 4.1 resume a semântica das lógicas apresentadas nessa seção.

Uma importante propriedade das lógicas que admitem ao mesmo tempo papéis transitivos e hierarquia de papéis é a capacidade de reduzir polinomialmente o problema da satisfatibilidade de um conceito D em relação a um TBox $\mathcal{T}$ (com axiomas GCI) ao problema de decidir a satisfatibilidade de um conceito D' em relação ao TBox vazio [HST00]. Essa propriedade é conhecida como

[4] Essa restrição é necessária para manter a decidibilidade [HST00].

tipo de dado (**D**)	D	$D^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}_{\mathbf{D}}$
papel concreto	U	$U^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}_{\mathbf{D}}$
valor de dado	v	$v^{\mathcal{I}} \in \Delta^{\mathcal{I}}_{\mathbf{D}}$
restrição existencial	$\exists U.D$	$\{x : \exists (x,y) \in U^{\mathcal{I}} \wedge y \in D^{\mathcal{I}}\}$
restrição de valor	$\forall U.D$	$\{x : \forall (x,y) \in U^{\mathcal{I}} \rightarrow y \in D^{\mathcal{I}}\}$
papéis funcionais ($\mathcal{F}$)	$\leq_1 R$	$\{x : \#\{(x,y) \in R^{\mathcal{I}}\} \leq 1\}$
	$\leq_1 U$	$\{x : \#\{(x,y) \in U^{\mathcal{I}}\} \leq 1\}$
	$\geq_1 R$	$\{x : \#\{(x,y) \in R^{\mathcal{I}}\} \geq 1\}$
	$\geq_1 U$	$\{x : \#\{(x,y) \in U^{\mathcal{I}}\} \geq 1\}$
restrição numérica ($\mathcal{N}$)	$\leq_n R$	$\{x : \#\{(x,y) \in R^{\mathcal{I}}\} \leq n\}$
	$\leq_n U$	$\{x : \#\{(x,y) \in U^{\mathcal{I}}\} \leq n\}$
	$\geq_n R$	$\{x : \#\{(x,y) \in R^{\mathcal{I}}\} \geq n\}$
	$\geq_n U$	$\{x : \#\{(x,y) \in U^{\mathcal{I}}\} \geq n\}$
restr. num. qualif. ($\mathcal{Q}$)	$\leq_n R.C$	$\{x : \#\{(x,y) \in R^{\mathcal{I}} \wedge y \in C\} \leq n\}$
	$\leq_n U.D$	$\{x : \#\{(x,y) \in U^{\mathcal{I}} \wedge y \in D\} \leq n\}$
	$\geq_n R.C$	$\{x : \#\{(x,y) \in R^{\mathcal{I}} \wedge y \in C\} \geq n\}$
	$\geq_n U.D$	$\{x : \#\{(x,y) \in U^{\mathcal{I}} \wedge y \in D\} \geq n\}$
nominal ($\mathcal{O}$)	$\{a_1, \ldots, a_n\}$	$\{d : d = a_i^{\mathcal{I}}\}$
papel inverso ($\mathcal{I}$)	R^-	$\{(x,y) : (y,x) \in R^{\mathcal{I}}\}$
papel transitivo ($\mathcal{S}$)	$R \in R_+$	$\{R^{\mathcal{I}} = (R^{\mathcal{I}})^+\}$
hierarquia de papéis ($\mathcal{H}$)	$R \sqsubseteq S$	$R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$

Figura 4.1: Extensões de $\mathcal{ALC}$.

internalização do TBox. As lógicas que admitem papéis inversos e nominais também possuem essa propriedade [Tob00].

Como os problemas de inferência em $\mathcal{ALC}$ com relação a um TBox genérico é ExpTime-completo, concluímos que as tarefas de inferência nas lógicas que admitem papéis transitivos e hierarquia de papéis, mesmo com TBox vazio, é ExpTime-difícil. O mesmo vale para lógicas que admitem papéis inversos junto de nominais.

As Figuras 4.2 e 4.3 resumem as classes de complexidade para de diversas DLs[5].

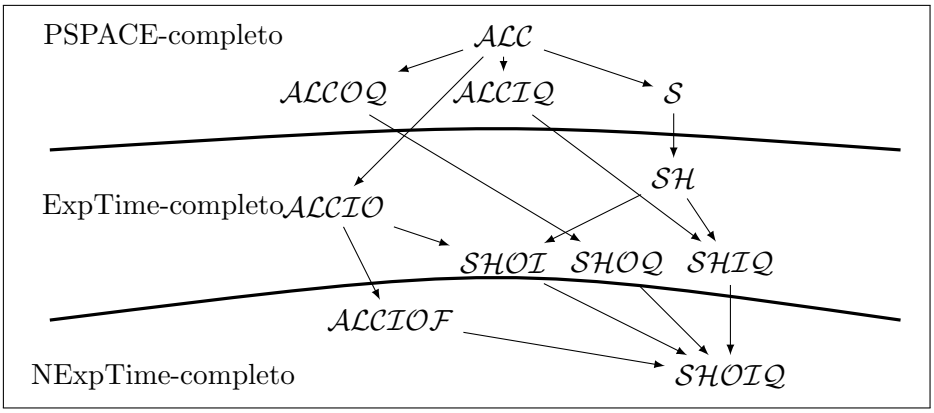


Figura 4.2: Complexidade das LDs sem axiomas GCI.

4.3 OWL

Em 2004 a W3C[6] anunciou oficialmente a adoção da linguagem OWL como padrão para representação de ontologias na web. A linguagem OWL deveria ser capaz de representar informações

[5] As informações usadas para construir essa figura foram tiradas de `http://www.cs.man.ac.uk/~ezolin/dl`.

[6] `http://www.w3c.org`

sobre categorias de objetos e como esses objetos se relacionam. Além disso, essa linguagem deveria ser compatível com outros padrões da web como XML, RDF e RDFS.

Como argumentamos na seção anterior, as lógicas de descrição são bons formalismos para representar ontologias. O estudo das lógicas de descrição levou a um entendimento profundo das características que as tornam indecidíveis, o que possibilitou explorar os fragmentos decidíveis mais expressivos. Além disso, no fim dos anos noventa e começo dos anos dois mil foram desenvolvidos diversos motores de inferência eficientes para lógicas de descrição bastante expressivas como o RACER [HM01] e o FaCT [Hor98].

A primeira versão de OWL [DSB+04] era formada por três linguagens. A mais geral foi chamada de OWL 1 Full e é completamente compatível com a linguagem RDFS, porém os problemas de inferência associados a essa linguagem são indecidíveis.

A linguagem OWL 1 DL é uma sub-linguagem de OWL 1 Full cujos problemas de inferência são decidíveis. O OWL 1 DL é uma variação sintática da lógica de descrição $\mathcal{SHOIN}(\mathbf{D})$, ou seja, da lógica que estende $\mathcal{ALC}$ com papéis transitivos ($\mathcal{S}$), hierarquia de papéis ($\mathcal{H}$), nominais ($\mathcal{O}$), papéis inversos ($\mathcal{I}$), restrição numérica simples ($\mathcal{N}$) e tipos de dados ($\mathbf{D}$). Hoje em dia, apesar da alta complexidade dessa lógica, existem diversos motores de inferência eficientes para OWL 1 DL [TH06, MSH07, SPG+07].

A terceira sub-linguagem é a OWL 1 Lite, que restringia ainda mais a expressividade de OWL 1 Full. A lógica de descrição associada a lógica OWL 1 Lite é a lógica $\mathcal{SHIF}(\mathbf{D})$. A lógica $\mathcal{SHIF}(\mathbf{D})$ possui todas os construtores de $\mathcal{SHOIN}(\mathbf{D})$ menos os nominais e as restrições numérica $\leq_n R.\top$ ($\geq_n R.\top$) são restritas a $n = 1$. Como mostramos na seção 4.2, mesmo essa linguagem possui alta complexidade computacional (ExpTime-completo).

A segunda versão do OWL [OWL09] foi oficialmente adotada como padrão para representação de ontologias no fim de 2009. A linguagem OWL 2 DL é uma variação sintática de $\mathcal{SROIQ}$ [HKS06] que estende $\mathcal{SHOIN}$ com as seguintes novas funcionalidades[7]:

- papéis disjuntos. Assim podem impor que, por exemplo, *irmã* e *mãe* são papéis disjuntos para garantir que ninguém é ao mesmo tempo mãe e irmã de alguém.
- papéis reflexivos e irreflexivos. Assim podemos, por exemplo, impor que *conhece* é um papel reflexivo para garantir que todos conhecem a si mesmos.
- asserção de papéis negada. Assim podemos escrever $\neg gosto(eu, bolo)$, por exemplo para representar que eu não gosto de bolo.
- hierarquia complexa de papéis. Hierarquia complexa de papéis admite inclusões da forma $R \circ S \sqsubseteq R$ ou $S \circ R \sqsubseteq R$. Podemos dizer, por exemplo, que $dono \circ temParte \sqsubseteq dono$ para garantir que se alguém é dono de uma parte de algo então esse alguém é dono deste algo.

[7] Os exemplos abaixo foram tirados de [HKS06]

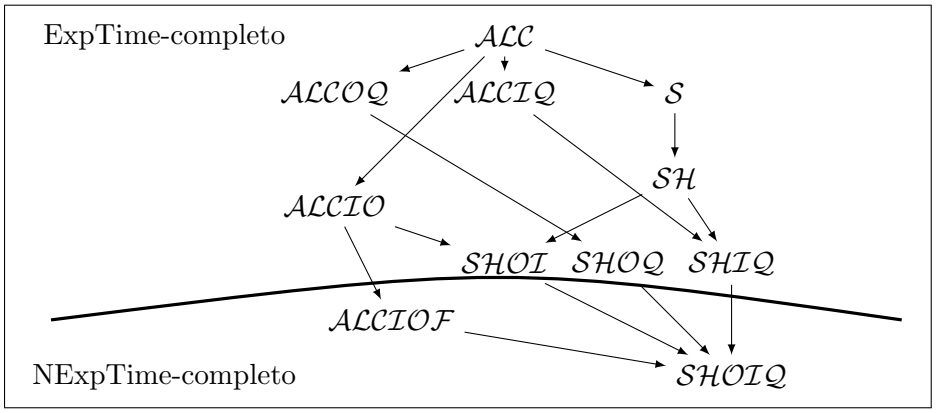


Figura 4.3: Complexidade das LDs com axiomas GCI.

- papel universal. O papel universal é o papel U em que $U^{\mathcal{I}} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ para toda interpretação $\mathcal{I}$.
- conceito $Self$. Com esse conceito podemos, por exemplo, definir narcisista como $gosta.Self$.

O padrão OWL 2 define também fragmentos tratáveis de OWL 2 DL, ou seja, sub-linguagens de OWL 2 DL cujos problemas de inferência podem ser resolvidos em tempo polinomial. Esses fragmentos de OWL 2 DL são chamados perfis OWL [MGH+08]. Foram definidos três desses perfis, cada um adequado para um tipo de problema:

1. *OWL EL:* adequado para representar ontologias que possuem uma grande quantidade de conceitos simples. Apesar da sua pouca expressividade, em OWL EL é possível modelar de forma satisfatória importantes ontologias médicas como SNOMED [SR04] e grande parte da ontologia GALEN [RRP96]. A lógica de descrição por trás desse perfil pertence a família das lógicas $\mathcal{EL}$ [BBL05].
2. *OWL QL:* indicada para ontologias com numero grande de instâncias cujo principal uso é responder consultas. Apesar da baixa expressividade, é possível modelar em OWL QL as principais características das linguagens UML e ER (entidade relacionamento). A lógica de descrição por trás desse perfil pertence a família das lógicas $\mathcal{DL}$-Lite [CGL+05, ACKZ09].
3. *OWL RL:* adequada para aplicações que alta eficiência computacional, mas sem sacrificar muito a expressividade. A eficiência da OWL RL vêm da possibilidade de implementá-la utilizando regras.

4.4 Relação com Outros Formalismos

As lógicas de descrição se relacionam com vários outros formalismos como seus precursores sistemas de frames [Min81] e redes semânticas [Qui67], a lógica de primeira ordem e diversas lógicas modais como a lógica K_n, as lógicas híbridas $\mathcal{H}_N$ e as lógicas dinâmicas (PDL). Nessa seção mostraremos a relação das LDs com a lógica de primeira ordem e com as lógicas modais K_n e $\mathcal{H}_N$.

4.4.1 Relação com a Lógica de Primeira Ordem

Lógicas de Descrição são fragmentos decidíveis da lógica de primeira ordem. Ou seja, é possível, como mostraremos neste capítulo, traduzir as sentenças de uma LD em sentenças de lógica de primeira ordem. Nessa seção mostraremos como traduzir conceitos e sentenças em LDs em fórmulas em lógica de primeira ordem.

Cada conceito em LD C pode ser traduzido em uma fórmula $\phi_C(x)$ com uma variável livre tal que para cada interpretação $\mathcal{I}$ os elementos de Δ que satisfazem $\phi_C(x)$ correspondem exatamente a $C^{\mathcal{I}}$.

$$
\begin{aligned}
\phi_{C \sqcap D}(x) &= \phi_C(x) \wedge \phi_D(x) \\
\phi_{C \sqcup D}(x) &= \phi_C(x) \vee \phi_D(x) \\
\phi_{\exists R.C}(y) &= \exists x R(y,x) \wedge \phi_C(x) \\
\phi_{\forall R.C}(y) &= \forall x R(y,x) \rightarrow \phi_C(x) \\
\phi_{\leq_n R.C}(x) &= \exists y_1, \ldots, y_n R(x,y_1) \wedge \cdots \wedge R(x,y_n) \wedge \bigwedge_{i<j} y_i \neq y_j \\
\phi_{\geq_n R.C}(x) &= \forall y_1, \ldots, y_n R(x,y_1) \wedge \cdots \wedge R(x,y_n) \rightarrow \bigvee_{i<j} y_i = y_j
\end{aligned}
$$

Desta forma, uma sentença $C \sqsubseteq D$ é traduzida em uma fórmula fechada $\forall x(\phi_C(x) \rightarrow \phi_D(x))$ e um TBox $\mathcal{T} = \{C_1 \sqsubseteq C_2, \ldots, C_{k-1} \sqsubseteq C_k\}$ corresponde a fórmula $\bigwedge_{2 \leq i \leq k} \forall x(\phi_{C_{i-1}}(x) \rightarrow \phi_{C_i}(x))$. Por fim, temos que $\mathcal{T} \vDash D_1 \sqsubseteq D_2$ sse $\bigwedge_{2 \leq i \leq k} \forall x(\phi_{C_{i-1}}(x) \rightarrow \phi_{C_i}(x)) \vDash \forall x(\phi_{D_1}(x) \rightarrow \phi_{D_2}(x))$.

Podemos então considerar que as lógicas de descrição são variantes notacionais de fragmentos da lógica de primeira ordem. O uso de uma notação diferente para LDs se justifica por ser bem mais concisa em alguns casos (e.g. restrições numéricas).

4.4.2 Relação com Lógicas Modais

Existe uma relação íntima entre lógicas de descrição e determinadas lógicas modais. Nessa seção mostraremos que o operador de consequência da lógica $\mathcal{ALC}$ com ABox vazio é equivalente ao operador de consequência global para a lógica multimodal K_n. Além disso mostraremos o operador de consequência para $\mathcal{ALCO}$ é equivalente ao operador de consequência para a lógica híbrida $\mathcal{H}_N$.

A Lógica Multimodal K_n

Aproveitaremos a ocasião para definir a lógica multimodal K_n e mostrar algumas propriedades que ela satisfaz.

Como fizemos com LPC e com as lógica $\mathcal{ALC}$ vamos começar definindo a linguagem de K_n. A assinatura de K_n consiste de dois conjuntos enumeráveis o primeiro é um conjunto de variáveis proposicionais $\mathbb{P}$ e o segundo é um conjunto de relações $\mathbb{R}$. A linguagem $\mathscr{L}_{K_n}$ é definida pela seguinte gramática em que $p \in \mathbb{P}$ e $r \in \mathbb{R}$.

$$\alpha := \ p \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \alpha \rightarrow \alpha \mid \neg\alpha \mid [r]\alpha \mid \langle r\rangle\alpha$$

Note que a linguagem da lógica modal K_n é uma extensão da linguagem da LPC.

Um modelo (de Kripke) $\mathcal{M}$ na lógica K_n é uma tupla $\langle \mathbf{V}, \mathbf{W}, \mathbf{r_1}, \ldots\rangle$ em que $\mathbf{W}$ é um conjunto não vazio cujos elementos são chamados de *mundos possíveis*, $\mathbf{V} : \mathbb{P} \rightarrow 2^{\mathbf{W}}$ é uma função chamada *valoração* que leva sentenças a conjuntos de mundos possíveis (intuitivamente àqueles mundos que em que vale α) e $\mathbf{r_i} \in \mathbb{R}$ é uma relação binária nos mundos $\mathbf{r_i} \subseteq \mathbf{W} \times \mathbf{W}$ chamada *relação de acessibilidade*. Um modelo $\mathcal{M}$ satisfaz uma sentença α em um mundo $w \in \mathbf{W}$ (escrevemos $w \vDash_{\mathcal{M}} \alpha$) sse:

$w \vDash_{\mathcal{M}} p$ sse $w \in \mathbf{V}(p)$

$w \vDash_{\mathcal{M}} \neg\alpha$ sse $w \nvDash_{\mathcal{M}} \alpha$

$w \vDash_{\mathcal{M}} \alpha \wedge \beta$ sse $w \vDash_{\mathcal{M}} \alpha$ e $w \vDash_{\mathcal{M}} \beta$

$w \vDash_{\mathcal{M}} \alpha \vee \beta$ sse $w \vDash_{\mathcal{M}} \alpha$ ou $w \vDash_{\mathcal{M}} \beta$

$w \vDash_{\mathcal{M}} \alpha \rightarrow \beta$ sse $w \nvDash_{\mathcal{M}} \alpha$ ou $w \vDash_{\mathcal{M}} \beta$

$w \vDash_{\mathcal{M}} \langle r\rangle\alpha$ sse existe w' tal que $(w, w') \in r$ e $w' \vDash_{\mathcal{M}} \alpha$

$w \vDash_{\mathcal{M}} [r]\alpha$ sse para todo w' se $(w, w') \in r$ então $w' \vDash_{\mathcal{M}} \alpha$

Dizemos que uma sentença α é válida em um modelo $\mathcal{M}$ (escrevemos $\vDash_{\mathcal{M}} \alpha$) sse para todo $w \in \mathbf{W}$ temos $w \vDash_{\mathcal{M}} \alpha$ e α é válida (escrevemos $\vDash \alpha$) sse α é válida em qualquer modelo $\mathcal{M}$.

Em lógicas modais temos duas formas de consequência: local (que denotaremos por $B \vDash^{loc} \alpha$ ou $\alpha \in C_{loc}(B)$) e global (que denotaremos por $B \vDash^{gl} \alpha$ ou $\alpha \in C_{gl}(B)$). Uma sentença α é consequência local de $B \in 2^{\mathscr{L}}$ sse para todo mundo $w \in W$ temos que $w \vDash_{\mathcal{M}} B$ (para todo $\beta \in B$, $w \vDash_{\mathcal{M}} \beta$) implica $w \vDash_{\mathcal{M}} \alpha$. Uma sentença α é consequência global de B se a validade de B em $\mathcal{M}$ implica a validade de α em $\mathcal{M}$. Note que para qualquer $\alpha \in \mathscr{L}$ temos $\vDash^{loc} \alpha$ sse $\vDash^{gl} \alpha$ sse $\vDash \alpha$. Ou seja, C_{loc} e C_{gl} possuem o mesmo conjunto de tautologias.

Em K_n, tanto C_{loc} quanto C_{gl} são tarskianas pelo mesmo motivo que LPC (Capítulo 2). Além disso, ambas são compactas e supra-clássicas. Porém, apenas C_{loc} satisfaz a dedução.

Exemplo 4.12: Com a consequência global temos que para qualquer $r \in \mathbb{R}$ vale $p \vDash_{gl} [r]p$, mas não vale $\vDash p \to [r]p$. O seguinte modelo $\mathcal{M} = \langle \mathbf{W}, \mathbf{V}, \mathbf{r} \rangle$ é um contra-exemplo para $\vDash p \to [r]p$. $\mathbf{W} = \{w_1, w_2\}$, $\mathbf{V}(p) = \{w_1\}$ e $\mathbf{r} = \{(w_1, w_2)\}$.

Schild mostrou em [Sch91] que a lógica $\mathcal{ALC}$ com ABox e TBox vazios é uma variante notacional da lógica K_n. Schild definiu uma tradução $.^t$ de conceitos de $\mathcal{ALC}$ em sentenças de K_n e provou que essa tradução preserva a satisfatibilidade. Apresentaremos aqui uma tradução ligeiramente diferente da apresentada por Schild. Vamos traduzir sentenças de $\mathcal{ALC}$ em sentenças de K_n:

$$\begin{aligned}
(\top \sqsubseteq A)^t &= p \\
(\top \sqsubseteq C \sqcup D)^t &= C^t \vee D^t \\
(\top \sqsubseteq C \sqcap D)^t &= C^t \wedge D^t \\
(\top \sqsubseteq \neg C)^t &= \neg C^t \\
(\top \sqsubseteq \forall R_i.C)^t &= [r_i]C^t \\
(\top \sqsubseteq \exists R_i.C)^t &= \langle r_i \rangle C^t
\end{aligned}$$

Conversamente podemos traduzir as sentenças de K_n em $\mathcal{ALC}$.

O resultado principal de Schild mostra que modelos de K_n podem ser traduzidos a modelos de $\mathcal{ALC}$ e vice-versa. Para isso basta tomar $\mathbf{W} = \Delta^{\mathcal{I}}$, $\mathbf{V}(C^t) = C^{\mathcal{I}}$ e $R_i^{\mathcal{I}} = \mathbf{r_i}$. Ou seja, interpretamos os mundos em K_n como indivíduos em $\mathcal{ALC}$ ou os indivíduos de $\mathcal{ALC}$ como mundos em K_n.

Desta tradução entre os modelos temos que uma sentença em $\mathcal{ALC}$ é válida, verdadeira em qualquer modelo $\mathcal{ALC}$, sse sua tradução em K_n é válida. Mais do que isso, temos que se uma sentença α em $\mathcal{ALC}$ é consequência de um conjunto de sentenças K (em símbolos $K \vDash \alpha$) então α^t em K_n é consequência global de K^t. Ou seja, existe uma equivalência entre os operadores de consequência: Seja Σ uma base de conhecimento em $\mathcal{ALC}$ com ABox vazio $\Sigma = \langle \mathcal{T}, \emptyset \rangle$ então $C_{\mathcal{ALC}}(\Sigma) = C_{gl}(\Sigma^t)$.

Exemplo 4.13: Em K_n temos que $\nvDash p \to [r]p$ (de forma equivalente $\nvDash \neg p \vee [r]p$), pois existe pelo menos um modelo que não satisfaz $p \to [r]p$ (mostrado no exemplo 4.12). Ainda em K_n temos que $p \nvDash [r]p$, pois existe um modelo em que p é válido, mas $[r]p$ não é válido.

Em $\mathcal{ALC}$ temos, então, que $\nvDash C \sqsubseteq \forall R.C$ (ou equivalentemente $\nvDash \top \sqsubseteq \neg C \sqcup \forall R.C$), pois podemos traduzir o modelo $\mathcal{M}$ em um modelo para que não satisfaz $C \sqsubseteq \forall R.C$: $\mathcal{I} = \langle \Delta^{\mathcal{I}}, .^{\mathcal{I}} \rangle$ tal que $\Delta = \{w, w'\}$, $C^{\mathcal{I}} = \{w\}$ e $R^{\mathcal{I}} = \{(w, w')\}$. Além disso, como todo modelo de K_n pode ser traduzido em um modelo para $\mathcal{ALC}$, temos que $\top \sqsubseteq C \vDash \top \sqsubseteq \forall R.C$.

A característica diferencial do ABox em termos de lógica modal é a capacidade de falar sobre um mundo (um indivíduo) em particular. As lógicas modais chamadas híbridas [Are00] foram desenvolvidas para lidar exatamente com esse problema. Para finalizar esta seção apresentaremos a lógica híbrida mais simples $\mathcal{H}_N$ e mostraremos como traduzir o operador de consequência de $\mathcal{ALCO}$ no operador de consequência global de $\mathcal{H}_N$.

A Lógica Híbrida $\mathcal{H}_N$

A assinatura da linguagem da lógica $\mathcal{H}_N$ possui, além do conjunto de variáveis proposicionais e de relações de acessibilidade, um conjunto de *nominais* $\mathbb{N}$. A linguagem de $\mathcal{H}_N$ simplesmente estende a gramática de K_n com elementos de $\mathbb{N}$. Estendemos também a valoração $\mathbf{V}$ para nominais ($\mathbf{V} : \mathbb{P} \cup \mathbb{N} \to 2^{\mathbf{W}}$) e temos que se $a \in \mathbb{N}$ então $\#\mathbf{V}(a) = 1$. Ou seja, a valoração $\mathbf{V}$ associa um nominal a um único mundo.

Exemplo 4.14: Seja $\mathcal{M} = \langle \mathbf{W}, \mathbf{V}, r \rangle$ um modelo, $w \vDash_{\mathcal{M}} a \to \langle r \rangle p$ sse $\mathbf{V}(a) = \{w\}$ e existe algum $w' \in \mathbf{W}$ tal que $(w, w') \in r$ e $w' \in \mathbf{V}(p)$

Relembramos que em $\mathcal{ALCO}$ qualquer sentença, mesmo do ABox, pode ser posta na forma $\top \sqsubseteq C$ para algum conceito C. Resta então mostrar como traduzir $\top \sqsubseteq \{a_1, \ldots, a_n\}$. Sejam $a_1, \ldots, a_n \in \mathbb{N}$

$$\top \sqsubseteq \{a_1, \ldots, a_n\}^t \quad = \quad a_1 \vee \cdots \vee a_n$$

Exemplo 4.15: A sentença $C(a)$ é equivalente a sentença $\top \sqsubseteq \neg\{a\} \sqcup C$ em $\mathcal{ALCO}$ que, por sua vez é traduzida para a sentença $a \to C^t$ em $\mathcal{H}_N$. De forma similar, $R(a, b)$ é traduzida para a sentença $a \to \langle r \rangle b$.

A tradução entre modelos de $\mathcal{H}_N$ e $\mathcal{ALCO}$ é a mesma que apresentamos para $\mathcal{ALC}$ e dela concluímos que o operador de consequência para $\mathcal{ALCO}$ é equivalente ao operador de consequência global para $\mathcal{H}_N$.

Outros construtores em LDs se relacionam com lógicas modais (e.g. papéis transitivos e a lógica $S4_m$, restrições numéricas e as modalidades graduadas [FBdC85]). A apresentação dessas lógicas está fora do escopo do nosso trabalho.

No capítulo seguinte definiremos propriedades lógicas e voltaremos a tratar das lógicas de descrição mostrando quais propriedades elas satisfazem.

Capítulo 5

Lógicas não Clássicas

O foco principal deste trabalho é estudar revisão de crenças em lógicas não clássicas (principalmente lógicas de descrição). Nossa abordagem consiste em estudar propriedades gerais de uma determinada classe de lógicas e então verificar que resultados de revisão de crenças valem em cada uma dessas classes.

Uma lógica, em termos gerais, foi definida como um par $\langle \mathscr{L}, Cn \rangle$. O conjunto $\mathscr{L}$ é a *linguagem* da lógica i.e. o conjunto de suas *fórmulas bem formadas*. O *operador de consequência* Cn é uma função ($Cn : 2^{\mathscr{L}} \to 2^{\mathscr{L}}$) que leva conjunto de fórmulas em suas consequências lógicas.

Começaremos esse capítulo motivando o estudo da teoria AGM em lógicas clássicas (seção 5.1). Para isso definiremos o conceito de compatibilidade AGM introduzido em [Flo06]. Em seguida estudaremos propriedades do operador Cn, *propriedades lógicas*, e como elas se relacionam entre si (seção 5.3). Por fim, mostramos para uma lista de lógicas que nos interessam quais dessas propriedades são satisfeitas (seção 5.4). A Seção 5.3 deste capítulo inaugura as nossas contribuições a área. Os resultados e exemplos apresentados a partir dessa seção são de nossa autoria (caso o texto não explicite o contrário).

5.1 Compatibilidade AGM

No capítulo 3.2 apresentamos a teoria AGM de revisão de crenças. Mostramos que uma das principais contribuições da área é a definição de conjuntos de postulados de racionalidade para as operações de contração e revisão em conjuntos de crenças e a demonstração dos teoremas de representação que mostram que esses postulados são equivalentes a determinada construção (e.g. partial meet). Este resultado vale para lógicas que satisfazem certas propriedades chamadas de suposições AGM. Nem toda lógica, porém, satisfaz essas suposições.

Em [FPA05] os autores argumentam que os postulados de racionalidade para a contração AGM não são nem aplicáveis a determinadas lógicas. Essas lógicas, chamadas de AGM incompatíveis, incluem as lógicas de descrição $\mathcal{SHIF}(\mathbf{D})$ e $\mathcal{SHOIN}(\mathbf{D})$. Nessa seção vamos apresentar as principais contribuições de [Flo06]. Definiremos formalmente o conceito de compatibilidade AGM e mostraremos a propriedade característica que determina se uma lógica é AGM-compatível (decomponibilidade).

Os postulados AGM para contração foram concebidos para serem aplicados a lógicas que satisfazem as suposições AGM. Ao tentar aplicar os resultados da teoria AGM a lógicas de descrição, os autores de [FPA05] notaram que os postulados AGM não são aplicáveis a qualquer lógica. Considere o seguinte exemplo:

Exemplo 5.1: [Flo06] Seja $\langle \mathscr{L}, Cn \rangle$ uma lógica tarskiana tal que:

$$\begin{aligned} \mathscr{L} &= \{a, b\} \\ Cn(b) = Cn(\mathscr{L}) &= \mathscr{L} \\ Cn(a) &= \{a\} \\ Cn(\emptyset) &= \emptyset \end{aligned}$$

Nessa lógica, se considerarmos $K = \mathscr{L}$ e $A = \{a\}$ temos que pela *inclusão*, pelo *sucesso* e pelo *fecho* a única possibilidade para $K - A$ é $\emptyset$. Porém, se $K - A = \emptyset$ temos que $K - A + A = Cn(a) = \{a\} \neq K$. Ou seja, a *recuperação* não é satisfeita.

O exemplo acima mostra uma lógica tarskiana em que não existe nenhuma contração $-$ para K que satisfaça todos os postulados AGM. A lógica acima é chamada de AGM incompatível. Formalmente:

Definição 5.2 (compatibilidade AGM [Flo06])**.** *Considere um conjunto de crenças K. Um operador $-$ em K é AGM-compatível sse $-$ satisfaz os seis postulados AGM básicos para contração em conjuntos de sentenças. Uma lógica $\langle \mathscr{L}, Cn \rangle$ é dita* AGM compatível *sse para todo $K \in 2^{\mathscr{L}}$ existe um operador $-$ em K que satisfaça todos os postulados AGM.*

A compatibilidade AGM está associada a existência de *conjuntos complementares*:

Definição 5.3 (conjuntos complementares [Flo06])**.** *Seja $\langle \mathscr{L}, Cn \rangle$ uma lógica tarskiana e sejam $K, A \in 2^{\mathscr{L}}$ dois conjuntos de sentenças tal que A é finitamente representável e $Cn(\emptyset) \subset Cn(A) \subset Cn(K)$. O* complemento de A com relação a K *(escrevemos $A^-(K)$) é a classe de conjuntos $K' \in 2^{\mathscr{L}}$ tal que $Cn(K') \subset Cn(K)$ e $Cn(K' \cup A) = Cn(K)$.*

Intuitivamente $A^-(K)$ é formado pelos conjuntos mais fracos que K que junto a A formam um conjunto equivalente a K. A existência de conjuntos complementares é a propriedade lógica que garante a compatibilidade entre a lógica e os postulados AGM. Lógicas que possuem essa propriedade são chamadas de *decomponíveis*:

decomponibilidade: [Flo06] Uma lógica $\langle \mathscr{L}, Cn \rangle$ é *decomponível* sse para todo $K, A \in 2^{\mathscr{L}}$ tal que A é finitamente representável e $Cn(\emptyset) \subset Cn(A) \subset Cn(K)$ o conjunto de complementos de A com relação a K não é vazio ($A^-(K) \neq \emptyset$). Ou seja, existe K' tal que $Cn(K') \subset K$ e $Cn(K' \cup A) = Cn(K)$.

Teorema 5.4. *[Flo06] Uma lógica $\langle \mathscr{L}, Cn \rangle$ é AGM compatível sse $\langle \mathscr{L}, Cn \rangle$ for decomponível.*

Na Seção 5.4 apresentaremos uma lista de lógicas não decomponíveis e que, portanto, não são compatíveis com a contração AGM.

5.2 Reticulados como Heurísticas

Para facilitar a visualização de seus resultados, Flouris [Flo06] usa o isomorfismo entre reticulados completos e lógicas tarskianas. Um *reticulado completo* é formado por um conjunto e uma relação de ordem parcial em que todo subconjunto possui um supremo e um ínfimo.

Definição 5.5 (ordem parcial)**.** *Uma relação $\leq$ é uma* ordem parcial *para P sse $\leq$ é uma relação em P transitiva (para todo $a, b, c \in P$ temos que $a \leq b$ e $b \leq c$ implica que $a \leq c$), reflexiva (para todo $a \in P$ temos $a \leq a$) e anti-simétrica (para todo $a, b \in P$ temos que $a \leq b$ e $b \leq a$ implica que $a = b$).*

Definição 5.6 (supremo e ínfimo)**.** *Seja $\leq$ uma ordem parcial e seja $H \subseteq P$. O elemento $sup(H)$ é o* supremo *de H sse para todo $a \in P$ temos que $b \leq a$ para todo $b \in H$ então $sup(H) \leq a$. O elemento $inf(H)$ é o* ínfimo *de H sse para todo $a \in P$ temos $b \geq a$ para todo $b \in H$ então $inf(H) \geq a$.*

Definição 5.7 (reticulado completo)**.** *Um* reticulado completo *é uma estrutura $\langle P, \leq \rangle$ em que $\leq$ é uma ordem parcial e todo subconjunto H de P possui um supremo e um ínfimo em P.*

Existe uma bijeção entre a classe dos reticulados completos e a classe das lógicas tarskianas. Ou seja, é possível associar um reticulado completo a cada lógica tarskiana e vice-versa.

Proposição 5.8. *[Flo06] Dada uma lógica tarskiana $\langle \mathscr{L}, Cn \rangle$ existe um reticulado completo $\langle P, \leq \rangle$ e uma função bijetora $f : 2^{\mathscr{L}} \to P$ tal que $f(B_1) \leq f(B_2)$ sse $Cn(B_1) \subseteq Cn(B_2)$.*

Está fora do escopo deste trabalho apresentar detalhes a teoria dos reticulados. Nos utilizaremos desse isomorfismo apenas para representar lógicas tarskianas de forma gráfica e como heurística para encontrar exemplos de lógicas tarskianas simples que satisfaçam (ou não satisfaçam) determinada propriedade. Considere a representação apresentada na figura 5.1 da lógica introduzida no exemplo 5.1. Nessa representação os nós representam conjuntos de crenças e os arcos representam a relação de inclusão entre eles.

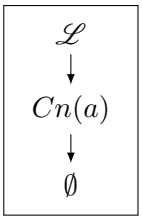


Figura 5.1: Exemplo de lógica não decomponível [Flo06].

Note que o supremo entre dois nós B_1 e B_2, o menor elemento que é maior que ambos, representa o conjunto $Cn(B_1 \cup B_2)$ e o ínfimo, maior elemento menor que ambos, representa o conjunto $Cn(B_1) \cap Cn(B_2)$.

Flouris mostra uma forma simples de identificar se uma lógica representada por um reticulado completo é decomponível. Basta verificar se para cada nó B do reticulado, a intersecção (ínfimo) dos elementos logo abaixo de B (os elementos maximais estritamente menores que B i.e. $B \perp B$) é $Cn(\emptyset)$ (o menor de todos). Essa propriedade é chamada de *corte máximo*. O reticulado da figura 5.1, por exemplo, não é decomponível, pois, a intersecção dos elementos maximais menores que $\mathscr{L}$ é $Cn(\alpha)$ e não $Cn(\emptyset)$.

Proposição 5.9. *[Flo06] Uma lógica $\langle \mathscr{L}, Cn \rangle$ é decomponível sse para todo conjunto de crenças K temos que $\bigcap K \perp K = Cn(\emptyset)$.*

5.3 Propriedades Lógicas

Nos capítulos anteriores definimos uma série de propriedades lógicas. Listaremos brevemente as propriedades que já foram definidas anteriormente:

Definição 5.10. *Para todo $B, B_1, B_2, B_3, A \in 2^{\mathscr{L}}$ e todo $\alpha, \beta \in \mathscr{L}$. A lógica $\langle \mathscr{L}, Cn \rangle$ satisfaz:*

monotonicidade: se $B_1 \subseteq B_2$ implica $Cn(B_1) \subseteq Cn(B_2)$.

idempotência: se $Cn(B) = Cn(Cn(B))$.

reflexividade: se $B \subseteq Cn(B)$.

tarskianicidade: se for monotônica, reflexiva e idempotente.

supraclassicalidade: se $\mathscr{L}$ é fechada pelos conectivos clássicos ($\wedge$, $\vee$, $\neg$ e $\rightarrow$) e se α pode ser inferido de B pela lógica proposicional clássica então $\alpha \in Cn(B)$.

dedução: se $\alpha \in Cn(B \cup \{\beta\})$ sse $(\beta \rightarrow \alpha) \in Cn(B)$.

compacidade: se $\alpha \in Cn(B)$ implica que existe um conjunto finito $B' \subseteq B$ tal que $\alpha \in Cn(B')$.

suposições AGM: se é tarskiana, supra-clássica, compacta e satisfaz a dedução.

explosão inconsistente: se A é inconsistente então $Cn(A) = \mathscr{L}$.

decomponibilidade: se $Cn(\emptyset) \subset Cn(A) \subset Cn(B)$ então existe A' tal que $Cn(A') \subset Cn(B)$ e $Cn(A \cup A') = Cn(B)$.

não contravenção local α: se $\neg\alpha \in Cn(B \cup \{\alpha\})$ então $\neg\alpha \in Cn(B)$

Nesse capítulo trataremos apenas de lógicas tarskianas. Portanto, toda vez que nos referirmos a uma lógica $\langle \mathscr{L}, Cn\rangle$, nos referimos a uma lógica tarskiana.

Como estamos tratando de lógicas tarskianas podemos, como mostramos na Seção 5.2, usar reticulados completos para gerar exemplos e para visualizar as lógicas.

A teoria AGM [AGM85] supõe que a linguagem da lógica subjacente ao problema da revisão seja fechada pelos conectivos lógicos usuais ($\wedge$, $\vee$, $\neg$ e $\rightarrow$). Lógicas de descrição, entre outras, não são fechadas por esses conectivos.

Por esse motivo consideraremos propriedades lógicas que não mencionam esses conectivos. As seguintes propriedades se mostraram centrais no estudo da revisão de crenças:

fecho por negação (clássica): Seja $X \in 2^{\mathscr{L}}$ um conjunto finitamente representável, dizemos que um conjunto finitamente representável $Y \in 2^{\mathscr{L}}$ é uma *negação (clássica)* de X sse $Cn(X \cup Y) = \mathscr{L}$ e $Cn(X) \cap Cn(Y) = Cn(\emptyset)$. Dizemos que uma lógica é *fechada por negação (clássica)* sse todo $X \in 2^{\mathscr{L}}$ finitamente representável possui pelo menos uma negação (clássica).

distributividade: Uma lógica $\langle \mathscr{L}, Cn\rangle$ é distributiva sse para todo $X, Y, Z \in 2^{\mathscr{L}}$ tal que Y e Z são finitamente representáveis, temos que $Cn(X \cup (Cn(Y) \cap Cn(Z))) = Cn(X \cup Y) \cap Cn(X \cup Z)$.[1]

Uma lógica $\langle \mathscr{L}, Cn\rangle$ que satisfaça essas duas propriedades é chamada de *booleana*.

Como no caso da decomponibilidade, podemos identificar se uma lógica é distributiva examinando o reticulado que a representa. Para isso precisamos definir *sub-reticulados*:

Definição 5.11 (sub-reticulado)**.** *Um* sub-reticulado *de um reticulado $\langle P, \leq\rangle$ é o par $\langle P_2, \leq_2\rangle$ em que $P_2 \subseteq P$ tal que $\leq_2$ é a restrição de $\leq$ em P_2 (i.e. se $B_1, B_2 \in P_2$ e $B_1 \leq B_2$ então $B_1 \leq_2 B_2$) e P_2 é fechado por supremo e ínfimo. Ou seja, se $B = sup(H)$ para $H \subseteq P_2$ então $B \in P_2$.*

Um reticulado é distributivo sse ele não possui nenhum sub-reticulado isomorfo ao reticulados $\mathfrak{m}_5$ nem ao reticulado $\mathfrak{n}_5$ [Gra71]. Os reticulados $\mathfrak{m}_5$ e $\mathfrak{n}_5$ estão representados na figura 5.3.

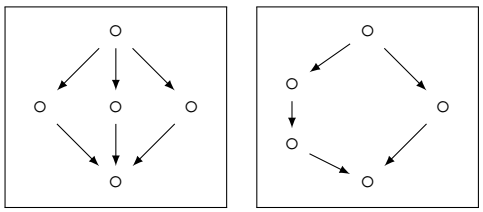
Por exemplo, a lógica apresentada no exemplo 5.1 é distributiva pois possui menos de cinco elementos e, logo, não pode possuir um sub-reticulado de cinco elementos. Já no exemplo 5.16 mostraremos uma lógica não distributiva, pois o sub-reticulado formado por $\mathscr{L}$, $Cn(c)$, $Cn(\{a, b\})$, $Cn(a)$ e $Cn(\emptyset)$ é isomorfo a $\mathfrak{n}_5$.

Tanto a distributividade quanto o fecho por negação seguem das suposições AGM:

Proposição 5.12. *Se uma lógica $\langle \mathscr{L}, Cn\rangle$ é supra-clássica e satisfaz dedução então ela é booleana.*

Demonstração. Veja o apêndice B.1. □

[1] Essa propriedade é uma generalização da seguinte propriedade: se $\beta \in Cn(B \cup \{\alpha_1\})$ e $\beta \in Cn(B \cup \{\alpha_2\})$ então $\beta \in Cn(B \cup \{\alpha_1 \vee \alpha_2\})$



(a) $\mathbf{m}_5$ (b) $\mathbf{n}_5$

Figura 5.2: Sub-reticulados que não ocorrem em lógicas distributivas.

Flouris em sua tese [Flo06] provou que lógicas booleanas são decomponíveis. Logo, como poderíamos imaginar, qualquer lógica que satisfaça as suposições AGM é AGM compatível.

Proposição 5.13. *[Flo06] Seja $\langle \mathscr{L}, Cn\rangle$ uma lógica booleana (distributiva e fechada por negação) então $\langle \mathscr{L}, Cn\rangle$ é decomponível.*

Esse resultado mostra que para uma lógica fechada por negação $\langle \mathscr{L}, Cn\rangle$, se $\langle \mathscr{L}, Cn\rangle$ for distributiva então ela é decomponível. A relação entre a negação clássica e a decomponibilidade é ainda mais profunda. Para mostrar isso precisamos definir o que são cadeias decrescentes.

Uma seqüência de conjuntos de sentenças $A_0, A_1, \ldots$ é chamada de *cadeia ascendente de conjuntos de sentenças* se satisfizer a seguinte propriedade: $Cn(A_0) \subset Cn(A_1) \subset \ldots$. Analogamente se $Cn(A_0) \supset Cn(A_1) \supset \ldots$ então chamamos a seqüência de *cadeia descendente de conjuntos de sentenças*. A presença de cadeias infinitas em uma lógica é uma propriedade que vale ser destacada:

propriedade da cadeia descendente (ascendente): Uma lógica $\langle \mathscr{L}, Cn\rangle$ satisfaz a *propriedade da cadeia descendente* (ascendente) sse toda cadeia descendente (ascendente) em $\langle \mathscr{L}, Cn\rangle$ é finita.

Proposição 5.14. *Lógicas decomponíveis que satisfaçam a condição da cadeia descendente são fechadas por negação.*

Demonstração. Veja o apêndice B.1. Aconselhamos a leitura dessa demonstração que esclarece bastante sobre as propriedades de uma lógica decomponível. □

Uma lógica é dita *finita* se ela admite um número finito de conjuntos de crenças distintos. Ou seja, se ela possui um número finito de classes de equivalência. É evidente que toda lógica finita satisfaz a propriedade da cadeia decrescente. Portanto, toda lógica finita e decomponível é fechada por negação.

No capítulo 3.3.2 mostramos uma propriedade para a negação chamada não contravenção local α. Essa propriedade é essencial para provar os resultados sobre revisão em bases de crenças [HW02]. Convém generalizarmos essa propriedade para conjuntos finitamente representáveis.

não contravenção local A: Seja A um conjunto finitamente representável, dizemos que A' satisfaz a não contravenção local em relação a A sse para qualquer $B \in 2^{\mathscr{L}}$, se $A' \subseteq Cn(B \cup A)$ então $A' \subseteq Cn(B)$

Em lógicas distributivas se um conjunto A possui uma negação clássica A', então A' satisfaz a não contravenção em relação a A:

Proposição 5.15. *Seja $\langle \mathscr{L}, Cn\rangle$ uma lógica distributiva. Se $Cn(A) \cap Cn(A') = Cn(\emptyset)$ então A' satisfaz a não contravenção em relação a A.*

Demonstração. Veja o apêndice B.1. □

Portanto, se uma lógica $\langle \mathscr{L}, Cn \rangle$ é booleana, então para todo A existe um A' que satisfaz a não contravenção local em relação a A. Portanto, como esperado, os resultados apresentados na Seção 3.3.2 são aplicáveis a qualquer lógica que satisfaça as suposições AGM.

Para finalizar essa seção usaremos representações gráficas para mostrar que determinadas propriedades lógicas são independentes. Já mostramos no exemplo 5.1 uma lógica distributiva e finita que não é decomponível. Portanto a decomponibilidade não segue da distributividade e da finitude.

A seguinte lógica decomponível finita não é distributiva e está representada na figura 5.3:

Exemplo 5.16:

$$\begin{aligned} \mathscr{L} &= \{a, b, c\} \\ Cn(\mathscr{L}) = Cn(\{b, c\}) = Cn(\{a, c\}) &= \mathscr{L} \\ Cn(\{a, b\}) &= \{a, b\} \\ Cn(a) &= \{a\} \\ Cn(b) &= \{b\} \\ Cn(c) &= \{c\} \\ Cn(\emptyset) &= \emptyset \end{aligned}$$

Note na figura 5.3 que a propriedade do corte máximo é satisfeita, logo essa lógica é decomponível. Considere o sub-reticulado gerado pelos elementos $\mathscr{L}$, $Cn(\{a, b\})$, $Cn(c)$, $Cn(a)$ e $Cn(\emptyset)$. Primeiro note que esses elementos formam um sub-reticulado, ou seja, esse conjunto é fechado por supremo e ínfimo. Agora repare que esse sub-reticulado é isomorfo a $\mathfrak{n}_5$. Logo, essa lógica é decomponível, finita, mas não é distributiva.

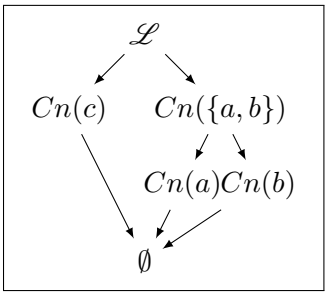


Figura 5.3: Lógica decomponível, finita e não-distributiva.

Por fim, o exemplo 5.17 mostra uma lógica booleana que não é compacta.

Exemplo 5.17:

$$\begin{aligned} \mathscr{L} &= \{a, x_0, x_1, \ldots, y_0, y_1 \ldots\} \\ Cn(a) &= \{y_0, y_1, \ldots\} \\ Cn(\{x_i : i \in I\} \cup \{y_j : j \in J\}) &= \{y_k : \exists i \in I(k \leq i) \text{ ou} \\ &\quad \exists j \in J(k \leq j)\} \cup \\ &\quad \{x_k : \exists i \in I(k \leq i)\} \text{ se } I \neq \emptyset \\ Cn(\emptyset) &= \emptyset \end{aligned}$$

Observando a figura 5.4 concluímos que essa lógica é distributiva, fechada por negação, logo, decomponível. Seja $Y = \{y_0, y_1, \ldots\}$, repare que $a \in Cn(Y)$, mas nenhum subconjunto finito de Y prova a. Portanto, essa lógica não é compacta.

O diagrama da figura 5.7 no fim desse capítulo resume os resultados apresentados nessa seção.

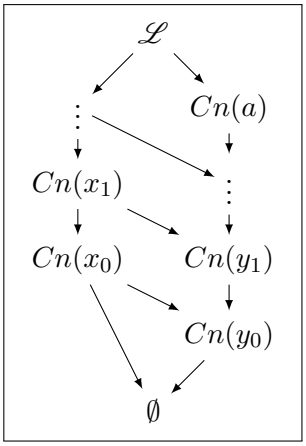


Figura 5.4: Lógica booleana e não compacta.

5.4 Algumas Lógicas não Clássicas

Nos propusemos a estudar a teoria AGM considerando apenas propriedades abstratas da lógica subjacente. Nessa seção iremos revisitar algumas lógicas conhecidas na literatura e verificar quais propriedades elas satisfazem.

Estamos particularmente interessados em lógicas que não satisfazem as suposições AGM. Nessas lógicas os resultados de [AGM85] não se aplicam. Mais drasticamente, nos interessam lógicas que não sejam AGM compatíveis.

Também nos interessa verificar quais lógicas são fechadas por negação, pois precisamos da negação para construir a revisão usando tanto a identidade de Levi ($K * A = K - \neg A + A$) quanto a identidade de Levi reversa ($B * A = B + A - \neg A$).

Escolhemos um conjunto de lógicas conhecidos da literatura para expor nosso ponto. Mostraremos que as lógicas LPC, K_n e H_N satisfazem as suposições AGM. As demais lógicas que analisaremos não satisfazem essas suposições. A lógica intuicionista (Seção 5.4.3) serve como exemplo de lógica distributiva, mas não decomponível. A lógica de Horn (secção 5.4.2) é um exemplo de lógica que não é nem distributiva nem fechada por negação. A lógica temporal LTL (Seção 5.4.4) foi escolhida por não ser compacta.

Além disso, na Seção 5.4.5 revisitaremos as lógicas de descrição. Mostraremos quais propriedades são satisfeitas por diversas lógicas de descrição.

5.4.1 Lógicas que Satisfazem as Suposições AGM

Nos capítulos anteriores mostramos algumas lógicas "bem comportadas", ou seja, lógicas que satisfazem as suposições AGM. Não são essas as lógicas que mais nos interessam, pois os resultados do capítulo 3 são perfeitamente aplicáveis a essas lógicas. Essas lógicas servem mais como contraponto para o estudo das demais lógicas estudadas nesse capítulo.

No capítulo 2 definimos a lógica proposicional clássica e mostramos que ela é tarskiana, compacta e que satisfaz a dedução. É evidente que a LPC é supra-clássica e, portanto como já esperávamos, LPC satisfaz as suposições AGM.

As lógicas modais K_n e $\mathcal{H}_N$ com consequência local também satisfazem as suposições AGM. Os operadores de consequência local dessas lógicas são tarskianos, compactos, supra-clássicos e satisfazem a dedução.

Nessas três lógicas os resultados AGM são aplicáveis sem alteração alguma. As demais lógicas que investigaremos nesse capítulo não satisfazem as suposições AGM.

5.4.2 Lógica de Horn

Nessa seção apresentaremos a lógica de Horn. A primeira parte dessa seção define e apresenta resultados conhecidos sobre a lógica de Horn. Na segunda parte mostraremos que lógicas de Horn são compactas e não são decomponíveis.

A lógica proposicional clássica (LPC) apesar de possuir expressividade bastante restrita possui alta complexidade computacional. O famoso teorema de Cook [Coo71] prova que o problema da satisfatibilidade em LPC é pelo menos tão complexo quanto qualquer problema NP. Em outras palavras, o problema da satisfatibilidade em LPC é *NP-completo*.

A lógica de Horn foi proposta como uma restrição da LPC cuja satisfatibilidade pode ser decidida em tempo polinomial. Apesar de possuir expressividade ainda mais restrita, a lógica de Horn tem um grande apelo computacional por sua baixa complexidade.

Diversas aplicações em inteligência artificial utilizam a lógica de Horn. A linguagem de programação PROLOG[2], por exemplo, é amplamente utilizada na área e é baseada na lógica de Horn.

A lógica de Horn $\langle \mathscr{L}_{Horn}, C_{Horn} \rangle$ é uma restrição da LPC em que a linguagem $\mathscr{L}_{Horn}$ é formada pelas seguintes sentenças α:

$$\begin{array}{rcl} h & = & p \\ b & = & \neg p \mid b \vee b \\ cl & = & h \mid b \mid h \vee b \\ \alpha & = & cl \mid \alpha \wedge \alpha \end{array}$$

Uma sentença em lógica de Horn é uma conjunção de cláusulas. Uma *cláusula* é uma disjunção de átomos eventualmente negados. As *cláusulas de Horn* possuem um *corpo* (disjunção de átomos negados) e podem possuir uma única *cabeça* (um átomo não negado). Assim, por exemplo, temos que $p_1 \vee \neg p_2, (p_1 \vee \neg p_2) \wedge (p_2 \vee \neg p_3), p, \neg p \in \mathscr{L}_{Horn}$, mas $p_1 \vee p_2 \notin \mathscr{L}_{Horn}$. A linguagem da lógica de Horn é um subconjunto próprio da linguagem da LPC ($\mathscr{L}_{Horn} \subset \mathscr{L}_{LPC}$).

O operador de consequência da lógica de Horn (C_{Horn}) é definido como a restrição de C_{LPC} à linguagem $\mathscr{L}_{Horn}$. O reticulado ilustrado na figura 5.5 representa a lógica de Horn cuja assinatura é $\mathbb{P} = \{p, q\}$.

Proposição 5.18. *A lógica de Horn $\langle \mathscr{L}, Cn \rangle$ não é decomponível.*

Demonstração. Veja a demonstração completa no apêndice B.1. □

Não é difícil notar também que a lógica de Horn é compacta.

Proposição 5.19. *A lógica de Horn é compacta.*

Por fim, temos que a lógica de Horn não é distributiva e não é fechada por negação como mostra o exemplo a seguir:

> **Exemplo 5.20:** Considere a lógica de Horn $\langle \mathscr{L}_{Horn}, C_{Horn} \rangle$ formada a partir da assinatura $\mathbb{P} = \{p, q\}$ (figura 5.5). A sentença $\neg p \wedge \neg q$ em $\langle \mathscr{L}_{Horn}, C_{Horn} \rangle$ não possui negação clássica, pois para qualquer $A \in 2^{\mathscr{L}_{Horn}}$ tal que $C_{Horn}(A) \neq C_{Horn}(\emptyset)$, temos que $C_{Horn}(A) \cap C_{Horn}(\neg p \wedge \neg q) \neq C_{Horn}(\emptyset)$ (verifique na figura que o ínfimo entre o nó $C_{Horn}(\neg p \wedge \neg q)$ e qualquer nó diferente de $C_{Horn}(\emptyset)$ é diferente de $C_{Horn}(\emptyset)$.).
>
> Além disso, note que $p \in C_{Horn}(\{p \vee \neg q, p\}) \cap C_{Horn}(\{p \vee \neg q, q\})$, mas $p \notin C_{Horn}(\{p \vee \neg q\} \cup (C_{Horn}(p) \cap C_{Horn}(q)))$. Logo, $\langle \mathscr{L}_{Horn}, C_{Horn} \rangle$ satisfaz a distributividade. Na figura 5.5 esse exemplo corresponde ao sub-reticulado formado por $C_{Horn}(p \vee \neg q)$, $C_{Horn}(p)$, $C_{Horn}(q)$, $C_{Horn}(p \wedge q)$ e $C_{Horn}(\emptyset)$ que é isomorfo a $\mathfrak{n}_5$.

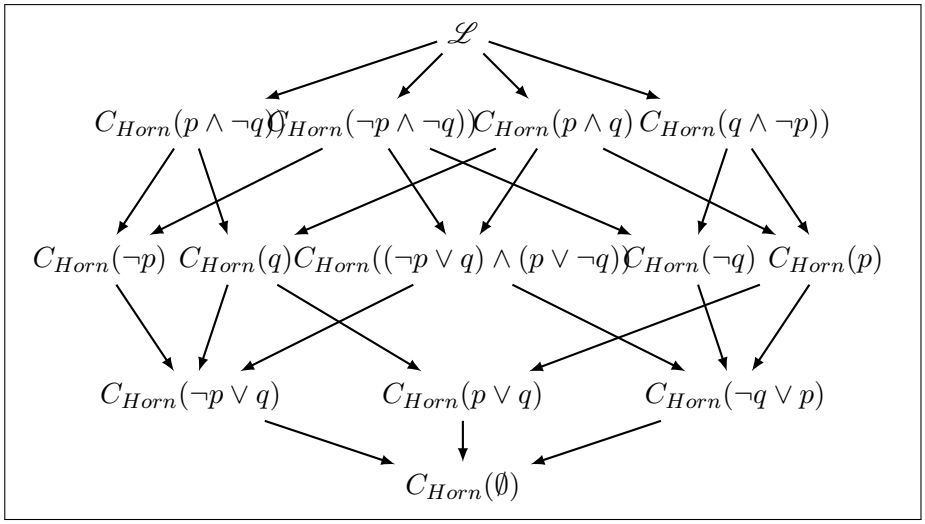


Figura 5.5: Lógica de Horn com assinatura $\mathbb{P} = \{p, q\}$

Resumindo, a lógica de Horn é compacta, mas não é nem decomponível, nem fechada por negação e nem distributiva.

Outros trabalhos que estudaram contração e revisão em lógica de Horn são [LSST08, DW10]. Em [LSST08] os autores mostram como descobrir se determinada fórmula em Horn possui complemento. Delgrande e Wassermann em [DW10] apresentam construções para contração em lógicas de Horn tanto em conjuntos de crenças quanto em bases.

5.4.3 Lógica Intuicionista

A lógica intuicionista é a lógica usada pelos matemáticos chamados construtivistas. A lógica intuicionista tem a mesma linguagem da lógica clássica, porém os conectivos, especialmente a negação, são interpretados de forma diferente. A lógica intuicionista foi concebida de forma a proibir a regra da eliminação da dupla negação e, conseqüentemente, a lei do terceiro excluído.

Vamos definir a semântica da lógica intuicionista a partir de modelos de Kripke. Um modelo de Kripke para a lógica intuicionista é uma tupla $\mathcal{M} = \langle \mathbf{W}, \mathbf{V}, \mathbf{R} \rangle$ em que $\mathbf{R}$ é uma relação transitiva. Ou seja, se $\{(w_1, w_2), (w_2, w_3)\} \subseteq \mathbf{R}$ então $(w_1, w_3) \in \mathbf{R}$. Temos então que:

$w \models_{\mathcal{M}} p$ sse $w \in \mathbf{V}(p)$

$w \models_{\mathcal{M}} \alpha \vee \beta$ sse $w \models \alpha$ ou $w \models \beta$

$w \models_{\mathcal{M}} \alpha \wedge \beta$ sse $w \models \alpha$ e $w \models \beta$

$w \models_{\mathcal{M}} \neg\alpha$ sse para todo w' tal que $(w, w') \in \mathbf{R}$, temos que $w' \not\models \alpha$

$w \models_{\mathcal{M}} \alpha \rightarrow \beta$ sse para todo w' tal que $(w, w') \in \mathbf{R}$, temos que se $w' \models \alpha$ então $w' \models \beta$

A lógica intuicionista não é AGM compatível. Em particular, não existe na lógica intuicionista complemento para p em relação a $p \vee \neg p$, ou seja, $\{p \vee \neg p\}^-(p) = \emptyset$.

Proposição 5.21. *A lógica intuicionista não é decomponível.*

Demonstração. Veja o apêndice B.1. Sugerimos ao leitor ler essa demonstração em particular. □

A figura 5.6 (reticulado de Rieger-Nishmura) representa o reticulado infinito para a lógica intuicionista com assinatura $\mathbb{P} = \{p\}$. Esse reticulado sugere que a lógica seja distributiva, pois não possui sub-reticulados isomorfos a $\mathfrak{m}_5$ ou a $\mathfrak{n}_5$. De fato temos que:

[2] http://www.swi-prolog.org/

Proposição 5.22. *A lógica intuicionista é distributiva.*

Demonstração. Segue trivialmente da regra da introdução da disjunção intuicionista. □

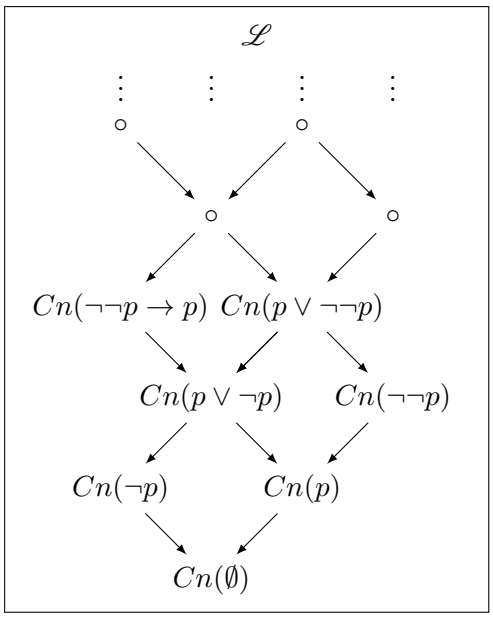


Figura 5.6: Reticulado de Rieger-Nishmura

Como corolário dos resultados acima temos que, como poderíamos esperar, a lógica intuicionista não é fechada por negação clássica. Resumindo, a lógica intuicionista é compacta e distributiva, mas não é fechada por negação clássica e nem decomponível.

5.4.4 Lógica Temporal Linear (LTL)

Lógicas temporais são bastante usadas em computação para verificação de programas. A linguagem de LTL é definida pela seguinte gramática cuja assinatura é um conjunto $\mathbb{P}$ de variáveis proposicionais:

$$\alpha := p \mid \alpha \vee \alpha \mid \alpha \wedge \alpha \mid \alpha \to \alpha \mid \neg\alpha \mid F\alpha \mid G\alpha \mid X\alpha \mid \alpha U \alpha$$

Os novos símbolos $F\alpha$, $G\alpha$, $X\alpha$ e $\alpha U \beta$ são interpretados como: no futuro α, sempre no futuro α, no próximo momento α e α até que β. Sua semântica é definida usando uma estrutura de Kripke $\mathcal{M} = \langle \mathbf{W}, \mathbf{V}, \mathbf{R} \rangle$. Um caminho π em $\mathcal{M}$ é uma seqüência de mundos $(w_0, w_1 \dots)$. Representamos por $\pi_{i,j}$ a seqüência que começa em w_i e termina em w_j $(w_i, \dots, w_j)$ e por π_i a seqüência infinita que começa em w_i $(w_i \dots)$. Temos então que:

$\pi \vDash_{\mathcal{M}} p$ sse $w_0 \vDash_{\mathcal{M}} p$

$\pi \vDash_{\mathcal{M}} \alpha \vee \beta$ sse $\pi \vDash \alpha$ ou $\pi \vDash \beta$

$\pi \vDash_{\mathcal{M}} \alpha \wedge \beta$ sse $\pi \vDash \alpha$ e $\pi \vDash \beta$

$\pi \vDash_{\mathcal{M}} \neg\alpha$ sse $\pi \nvDash \alpha$

$\pi \vDash_{\mathcal{M}} \alpha \to \beta$ sse $\pi \vDash \neg\alpha$ ou $\pi \vDash \beta$

$\pi \vDash_{\mathcal{M}} G\alpha$ sse para todo i temos $\pi_i \vDash \alpha$

$\pi \vDash_{\mathcal{M}} X\alpha$ sse $\pi_1 \vDash \alpha$

$\pi \vDash_{\mathcal{M}} \beta U \alpha$ sse para algum i temos que $\pi_{0,i-1} \vDash \beta$ e $\pi_i \vDash \alpha$.

A lógica LTL é um bom exemplo de lógica não compacta. Considere o exemplo:

Exemplo 5.23: Seja $K = \{p, Xp, XXp, XXXp, \dots\}$, então $Gp \in C_{LTL}(K)$. Porém, nenhum subconjunto finito de K implica Gp.

A ausência da compacidade causa problemas na construção partial meet. Para garantir que $K -_\gamma \alpha$ satisfaz o sucesso, temos que se $\alpha \notin Cn(\emptyset)$ e $\alpha \in K$ então necessariamente $K \perp \alpha \neq \emptyset$. Ou seja, deve existir um subconjunto maximal de K que não implica α. Esse resultado segue da propriedade do limite superior para o caso particular em que $X = \emptyset$:

Proposição 5.24 (propriedade do limite superior [AM81])**.** *Se $X \subseteq K$ e $\alpha \notin Cn(X)$ então existe X' tal que $X \subseteq X' \in K \perp \alpha$.*

A propriedade do limite superior para lógicas compactas é uma consequência do lema de Tuckey (lema 2.4) que é equivalente ao axioma da escolha.

Sem a compacidade não garantimos que a propriedade do limite superior e, logo, não garantimos que $K \perp \alpha \neq \emptyset$ para $\alpha \notin Cn(\emptyset)$. Nesse caso, não garantimos o sucesso da contração. Considere o seguinte exemplo:

Exemplo 5.25: Seja $K = C_{LTL}(\{p, Xp, XXp, XXXp, \dots\})$, apesar de $Gp \notin C_{LTL}(\emptyset)$, temos que $K \perp Gp = \emptyset$, pois não existe um elemento maximal de K que não implica Gp. Assim, temos que $\gamma(K \perp \alpha) = \{K\}$ e $\bigcap\gamma(K \perp \alpha) = K$ e então a contração partial meet $\bigcap\gamma(K \perp \alpha)$ não satisfaz o sucesso.

Uma outra lógica temporal bastante estudada na literatura é a CTL ("Computational Tree Logic"). Como a LTL, a lógica CTL não é compacta. Revisão de crenças em CTL foi estudada por Sousa e Wassermann em [dSW07] e por Guerra e Wassermann em [GW10].

5.4.5 Lógicas de Descrição

A principal aplicação do trabalho desenvolvido por Flouris [Flo06] foi a demonstração de que as lógicas de descrição $\mathcal{SHIF}(\mathbf{D})$ e $\mathcal{SHOIN}(\mathbf{D})$ não são decomponíveis e consequentemente não são AGM compatíveis. Esse resultado segue do seguinte lema:

Lema 5.26. *[Flo06] Considere uma lógica $\langle \mathscr{L}, Cn\rangle$. Se existe algum conjunto $K \in 2^{\mathscr{L}}$ tal que $Cn(\emptyset) \subset Cn\{x \in \mathscr{L} : Cn(x) \subset Cn(K)\} \subset Cn(K)$ então $\langle \mathscr{L}, Cn\rangle$ não é decomponível.*

Em outras palavras, o lema garante que se para algum conjunto K de sentenças da lógica, a união de todos as sentenças x mais fracas que K não forem suficientes para provar K, então a lógica não é decomponível. Os autores mostram então que uma LD $\langle \mathscr{L}, Cn\rangle$ que admite restrição de valor $\forall R.C$, restrição existencial $\exists R.\top$ ou restrição numérica e hierarquia de papéis, mas não admite outros construtores de papéis não será decomponível.

Teorema 5.27. *[Flo06, FPA05] Seja $\langle \mathscr{L}, Cn\rangle$ uma LD cuja assinatura possui pelo menos dois papéis atômicos e um conceito atômico. Se $\langle \mathscr{L}, Cn\rangle$ admite hierarquia de papéis ($R \sqsubseteq S$) e admite pelo menos um dos seguintes construtores:*

- *restrição de valor ($\forall R.C$)*
- *restrição existencial ($\exists R.\top$)*
- *restrição numérica ($\leq_n R.\top$ ou $\geq_n R.\top$)*

porém, não admite construtores para papéis, então $\langle \mathscr{L}, Cn\rangle$ não é decomponível.

A demonstração desse teorema usa o lema 5.26 e prova que a união todas as sentenças mais fracas que $R \sqsubseteq S$ não é vazia (pelo menos $\forall R.A \sqsubseteq \forall S.A$ pertence a ela) e não é suficiente para provar $R \sqsubseteq S$[3].

Corolário 5.28. *Nenhuma lógica de descrição entre $\mathcal{ALCH}$ e $\mathcal{ALCHOIQ}$ e entre $\mathcal{SH}$ e $\mathcal{SHOIQ}$(**D**), inclusive as lógicas do OWL 1 ($\mathcal{SHIF}$(**D**) e $\mathcal{SHOIN}$(**D**)), é decomponível. Portanto, nenhuma dessas lógicas é AGM compatível.*

Flouris [Flo06] mostrou também que diversas LDs da família $\mathcal{ALC}$ são decomponíveis desde que consideremos o ABox vazio.

Como as lógicas de descrição são fragmentos da lógica de primeira ordem e como a lógica de primeira ordem é compacta, temos que as LDs são compactas.

Proposição 5.29. *Toda lógica de descrição que é equivalente a um subconjunto da lógica de primeira ordem é compacta.*

Demonstração. Veja o apêndice B.1. □

Sabemos que diversas lógicas de descrição não são decomponíveis. Pela proposição 5.13, ou essas lógicas não são distributivas ou não são fechadas por negação. Vamos analisar ambas as propriedades.

Começamos com a distributividade. De maneira informal sabemos que a distributividade está relacionada com a disjunção. Uma vez que as lógicas modais possuem disjunção "bem comportada", nossa intuição indica que as LDs que são equivalentes as lógicas modais, $\mathcal{ALC}$ sem ABox e $\mathcal{ALCO}$, devem ser distributivas. De fato, do resultado abaixo segue que ambas lógicas são distributivas:

Proposição 5.30. *Considere um LD $\langle \mathscr{L}, Cn\rangle$ tal que para qualquer sentença $\alpha \in \mathscr{L}$ existe $\alpha' \in \mathscr{L}$ tal que $Cn(\alpha) = Cn(\alpha')$ e α' tem a forma $\top \sqsubseteq C$ para algum C. Então $\langle \mathscr{L}, Cn\rangle$ é distributiva.*

Demonstração. Veja o apêndice B.1. □

Ainda como corolário da proposição acima temos que qualquer lógica de descrição entre $\mathcal{ALC}$ e $\mathcal{ALCIQ}$ e entre $\mathcal{S}$ e $\mathcal{SIQ}$ na ausência de ABox é distributiva. Além disso, qualquer lógica entre $\mathcal{ALCO}$ e $\mathcal{ALCOIQ}$ e entre $\mathcal{SO}$ e $\mathcal{SOIQ}$ é distributiva mesmo na presença de ABox.

O exemplo a seguir mostra que a presença de hierarquia de papéis faz com que a lógica não seja distributiva.

> **Exemplo 5.31:** Seja $X = \{R \sqsubseteq S_1, R \sqsubseteq S_2\}$, $Y = \{S_1 \sqsubseteq S_3\}$ e $Z = \{S_2 \sqsubseteq S_3\}$. Temos que $C_{\mathcal{ALCH}}(Y) \cap C_{\mathcal{ALCH}}(Z) = C_{\mathcal{ALCH}}(\emptyset)$. Logo, $R \sqsubseteq S_3 \notin C_{\mathcal{ALCH}}(X \cup (C_{\mathcal{ALCH}}(Y) \cap C_{\mathcal{ALCH}}(Z)))$, mas $R \sqsubseteq S_3 \in C_{\mathcal{ALCH}}(X \cup Y) \cap C_{\mathcal{ALCH}}(X \cup Z)$.

Desse exemplo temos que nenhuma lógica de descrição entre $\mathcal{ALCH}$ e $\mathcal{ALCHOIQ}$ ou entre $\mathcal{SH}$ e $\mathcal{SHOIQ}$**(D)** é distributiva. Em particular as lógicas por trás do OWL 1 ($\mathcal{SHIF}$(**D**) e $\mathcal{SHOIN}$(**D**)) não são distributivas.

Como no caso da decomponibilidade, a interação entre ABox e TBox na ausência de nominais leva a LD a não ser distributiva.

> **Exemplo 5.32:** Seja $X = \{a = b\}$, $Y = \{C(a)\}$ e $Z = \{C(b)\}$, então $C_{\mathcal{ALC}}(Y) \cap C_{\mathcal{ALC}}(Z) = C_{\mathcal{ALC}}(\emptyset)$. Portanto, $C(a) \notin C_{\mathcal{ALC}}(X \cup (C_{\mathcal{ALC}}(Y) \cap C_{\mathcal{ALC}}(Z)))$, mas $C(a) \in C_{\mathcal{ALC}}(X \cup Y) \cap C_{\mathcal{ALC}}(X \cup Z)$.

Desse exemplo temos que, na presença do ABox, nenhuma lógica entre $\mathcal{ALC}$ e $\mathcal{ALCIQ}$ ou entre $\mathcal{S}$ e $\mathcal{SIQ}$ é distributiva.

Passemos agora a negação. Dois papeis R e S não se relacionam se nem $R \sqsubseteq S$ nem $S \sqsubseteq R$. Se uma LD admite pelo menos os construtores $\sqcup$, $\sqcap$ e $\neg$ e sua assinatura possui uma infinidade de papéis que não se relacionam, então essa lógica não é fechada por negação. Curiosamente, a maior parte das lógicas de descrição que mencionamos até aqui não é fechada por negação.

[3] Veja [Flo06] para mais detalhes.

Proposição 5.33. *Considere a LD $\langle \mathscr{L}, Cn\rangle$ que admite os construtores $\neg$, $\forall$, $\sqcup$ e axiomas GCI no TBox. Se a assinatura de $\langle \mathscr{L}, Cn\rangle$ possui uma infinidade de papéis não relacionados, então $\langle \mathscr{L}, Cn\rangle$ não é fechada por negação.*

Demonstração. Veja o apêndice B.1. □

Esse resultado segue do seguinte lema:

Lema 5.34. *Sejam A e B dois conceitos tais que $\top \sqsubseteq A$ e $\top \sqsubseteq B$ não são tautologias e suponha que exista um papel R não relacionado com nenhum papel que ocorre em A ou em B. Então, $Cn(\top \sqsubseteq A) \cap Cn(\top \sqsubseteq B) \neq \emptyset$.*

Demonstração. Veja o apêndice B.1. Sugerimos essa demonstração ao leitor interessado. □

Assim, notamos que as lógicas que satisfazem a hipótese do lema não satisfazem a propriedade da cadeia descendente. Em particular as lógicas $\mathcal{ALC}$ sem ABox e $\mathcal{ALCO}$ não satisfazem a propriedade da cadeia descendente. Logo, o teorema 5.14 é consistente com o fato de as lógicas $\mathcal{ALC}$ sem ABox e $\mathcal{ALCO}$ não serem fechadas por negação, mas serem decomponíveis.

Também vale notar que a proposição acima implica que, com operador de consequência lógica global, as lógicas modais K_n e $\mathcal{H}_N$ não são fechadas por negação.

A última propriedade que analisaremos é a não contravenção. O exemplo abaixo mostra que em certas LDs para todo conjunto A existe um conjunto A' que satisfaz a não contravenção local em relação a A, porém, nesses casos cada sentença nessas lógicas possui um infinito número de conjuntos com essa propriedade.

> **Exemplo 5.35:** Em lógicas que admitem $\sqcup$ e $\neg$ podemos representar um conjunto de sentenças do TBox como uma sentença $\top \sqsubseteq A$. Não é difícil notar que se $\mathcal{I} \vDash \top \sqsubseteq A$ então $\mathcal{I} \nvDash \top \sqsubseteq \neg A$. Portanto, para qualquer conceito B temos que $\top \sqsubseteq \neg A \in Cn(\{\top \sqsubseteq B\} \cup \{\top \sqsubseteq A\})$ implica em $\top \sqsubseteq \neg A \in Cn(\top \sqsubseteq B)$.
>
> Porém, pelo mesmo motivo outras sentenças também satisfazem a não contravenção em relação a $\top \sqsubseteq A$, por exemplo, todas sentenças da forma $\top \sqsubseteq \exists R \ldots \exists R \neg A$.

Esse exemplo nos mostra que, apesar de ser possível definir um negação que satisfaça a não-contravenção na maioria das LDs, essa negação não seria única. Portanto, não é claro qual das possíveis negações deve ser usada, por exemplo, para construir a revisão.

lógicas	**distributiva**	**negação**	**decomponível**	**compacta**
LPC	sim	sim	sim	sim
K_n e $\mathcal{H}_N$ (consequência local)	sim	sim	sim	sim
K_n e $\mathcal{H}_N$ (consequência global)	sim	não	sim	sim
Horn	não	não	não	sim
Intuicionista	sim	não	não	sim
LTL	sim	sim	sim	não
todas entre $\mathcal{ALC}$ e $\mathcal{ALCIQ}$ (sem ABox)	sim	não	sim [Flo06]	sim
todas entre $\mathcal{S}$ e $\mathcal{SIQ}$ (sem ABox)	sim	não	sim [Flo06]	sim
todas entre $\mathcal{ALCO}$ e $\mathcal{ALCOIQ}$	sim	não	sim [Flo06]	sim
todas entre $\mathcal{SO}$ e $\mathcal{SOIQ}$	sim	não	sim [Flo06]	sim
todas entre $\mathcal{ALCH}$ e $\mathcal{ALCHOIQ}$	não	não	não [Flo06]	sim
todas entre $\mathcal{SH}$ e $\mathcal{SHOIQ}$	não	não	não [Flo06]	sim
$\mathcal{SHIF}(\mathbf{D})$ e $\mathcal{SHOIN}(\mathbf{D})$	não	não	não [Flo06]	sim

Tabela 5.1: Propriedades das lógicas

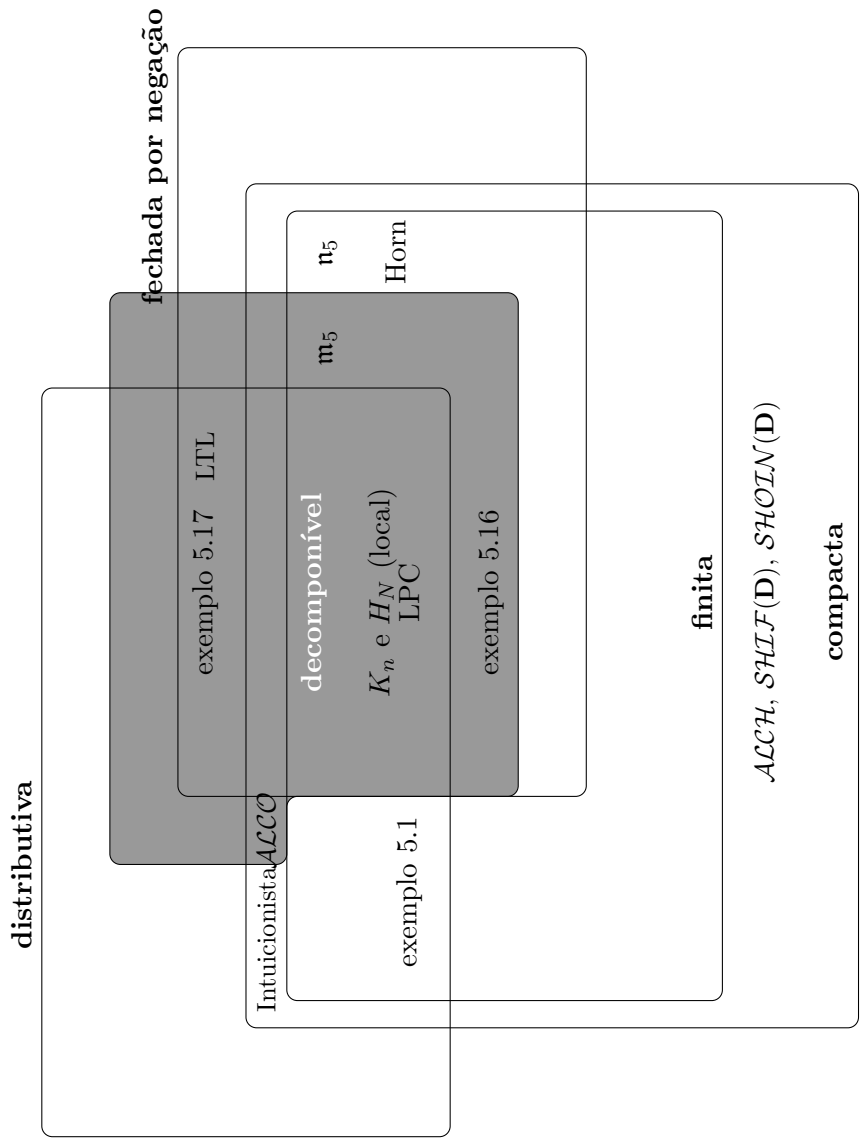


Figura 5.7: Topologia das propriedades das lógicas.

5.5 Conclusão

Nesse capítulo estudamos propriedades lógicas. Destacamos um conjunto de propriedades que julgamos úteis para o estudo da revisão de crenças (decomponibilidade, distributividade, compacidade etc.) e verificamos como elas se relacionam. Além disso, escolhemos algumas lógicas (lógica de Horn, lógica intuicionista, LTL etc.) para mostrar quais dessas propriedades elas satisfazem (veja a tabela 5.1).

A figura 5.7 resume os resultados desse capítulo. Os pontos nessa figura representam lógicas. A figura mostra, por exemplo, que a lógica LTL é distributiva, fechada por negação e não é compacta. Note que não existem lógicas finitas e decomponíveis que não sejam fechadas por negação, mas existem lógicas como $\mathcal{ALCO}$ que são decomponíveis e não são fechadas por negação. A figura mostra também que toda lógica booleana é decomponível.

Os resultados clássicos da teoria AGM assumem que a lógica subjacente satisfaça as suposições AGM. Como as suposições AGM mencionam os conectivos lógicos da LPC focamos nosso estudo em propriedades mais fracas do que as suposições AGM: distributividade, fecho por negação e compacidade. Grande parte das lógicas apresentadas nesse capítulo não satisfaz alguma dessas propriedades e portanto não satisfaz as suposições AGM.

Mostramos na Seção 5.1 que nem toda lógica é compatível com os postulados AGM para contração. Na Seção 5.4 mostramos várias lógicas incompatíveis com a contração AGM.

A revisão AGM, por sua vez, assume que a lógica subjacente seja fechada por negação. Mostramos que diversas lógicas que nos interessam não são fechadas por negação clássica.

Nos próximos capítulos mostraremos resultados na área de revisão de crenças que valem para lógicas AGM incompatíveis e para lógicas que não são fechadas por negação. A maior parte desses resultados vale para qualquer lógica compacta. Ou seja, com exceção da lógica LTL, esses resultados se aplicam as demais lógicas apresentadas nesse capítulo.

Capítulo 6

Contração AGM em Lógicas não Clássicas

Neste capítulo tratamos da contração em conjuntos de crenças. Nosso objetivo é mostrar como essa teoria se aplica a lógicas não clássicas.

Na Seção 5.1 mostramos que diversas lógicas não são compatíveis com os postulados AGM para contração. Na Seção 6.1 mostraremos como resolver este problema substituindo o postulado da recuperação pelo postulado da relevância. Esse novo conjunto de postulados é equivalente aos postulados AGM para lógicas booleanas, mas não para qualquer lógica tarskiana em geral (Seção 6.1.1). Além disso, esse conjunto de postulados é mais adequado para caracterizar a contração partial meet (Seção 6.1.2). Por fim, mostraremos que ele é compatível com uma classe maior de lógicas, todas as lógicas compactas (Seção 6.1.3)

6.1 Recuperação e Relevância em Lógicas não Clássicas

Dos postulados AGM o da recuperação é certamente o mais controverso. Desde as primeiras publicações na área de revisão de crenças vários trabalhos criticaram diversos aspectos da recuperação [Han91, Mak87, Nie91, Lev91]. Makinson em [Mak87] o questiona, por exemplo, argumentando que esse é o único postulado desnecessário para construir a revisão usando a identidade de Levi. Flouris [FPA06] alega que a incompatibilidade dos postulados AGM com diversas lógicas não clássicas é fruto da interação entre o postulado da recuperação e os demais postulados AGM. Hansson [Han91] afirma que a recuperação não é intuitiva em diversas situações.

Nessa seção focaremos nas duas últimas críticas ao postulado da recuperação. Essas críticas servirão de motivação para os resultados apresentados no restante desse capítulo em que argumentamos que o postulado da relevância, definido em [Han91], é de fato um bom substituto para o postulado da recuperação em lógicas não clássicas.

Em [Han91], Hansson argumenta que o postulado da recuperação é anti-intuitivo em diversas situações. O seguinte exemplo, tirado de [Han91], mostra uma dessas situações:

Exemplo 6.1:

> "Eu acreditava que 'Cleopatra tinha uma filho' (ϕ) e que 'Cleópatra tinha uma filha' (ψ), e logo que 'Cleópatra tinha uma criança' ($\phi \vee \psi$). Então eu recebo uma informação que me faz desistir da minha crença em $\phi \vee \psi$, e contrair meu conjunto de crenças de acordo, formando $K - (\phi \vee \psi)$. Logo em seguida eu aprendo por uma fonte confiável que Cleópatra tinha uma criança. Me parece perfeitamente razoável adicionar $\phi \vee \psi$ ao meu conjunto de crenças sem reintroduzir nem ϕ nem ψ."

Uma alternativa natural ao paradigma AGM para contração é o operação *withdrawal*. Essa operação, definida por Makinson em [Mak87], difere da contração AGM apenas pela ausência

do postulado da recuperação. Ou seja, uma operação $-$ para um conjunto de crenças K é um *withdrawal* se ela satisfaz todos os postulado AGM para contração exceto a recuperação. Porém, como mostramos no capítulo 3, a operação withdrawal não garante a minimalidade da mudança de forma satisfatória.

Por esse motivo, Hansson [Han91] defende que não podemos simplesmente abandonar a recuperação. Devemos substituir o postulado da recuperação por algum outro que garanta a minimalidade, mas que seja mais intuitivo. Para isso Hansson definiu o postulado da relevância [Han91]:

(relevância) Se $\beta \in K \setminus K - A$ então existe K' tal que $K - A \subseteq K' \subseteq K$, $A \nsubseteq K'$, mas $A \subseteq K' + \beta$.

O postulado da relevância garante a minimalidade da operação impedindo que sentenças irrelevantes sejam removidas. Ele impõe que nenhum elemento β possa ser removido de K a menos que β contribua para provar A, ou seja, para algum K' tal que $K - A \subseteq K' \subseteq K$ o conjunto $K' \cup \{\beta\}$ prova A. Uma operação de contração $-$ que satisfaz os postulados de withdrawal mais a relevância será chamada de *contração-relevante*.

Ainda em [Han91], Hansson provou que para lógicas que satisfazem as suposições AGM, a relevância e a recuperação são equivalentes na presença do outros postulados AGM. Ou seja, a contração AGM é equivalente a contração-relevante caso a lógica subjacente satisfaça as suposições AGM. Por esse motivo Hansson conclui que a recuperação deve ser aceita "como uma propriedade emergente, ao invés de como um postulado fundamental, da contração em conjuntos de crenças".

A crítica de Flouris ao postulado da recuperação consiste em mostrar que a incompatibilidade AGM em lógicas não clássicas é fruto da interação entre a recuperação e os demais postulados AGM.

Como vimos no capítulo 5, diversas lógicas de interesse computacional não são compatíveis com os postulados AGM. Para aplicar a teoria de revisão de crenças a essas lógicas temos que alterar de alguma forma os postulados AGM. Segundo Flouris devemos substituir o postulado da recuperação.

Flouris [FPA06] sugere que o postulado da recuperação seja substituído pela *recuperação ótima* [FPA06].

(recuperação ótima) Se $Cn((K - X) \cup X) \subset Cn(Y \cup X)$ para algum $Y \subseteq Cn(K)$, então $Cn(\emptyset) \subset Cn(X) \subseteq Cn(Y)$

Usaremos o termo *recuperação AGM* ao nos referir ao postulado da recuperação para distinguí-lo da recuperação ótima.

Neste capítulo aprofundaremos o estudo do postulado da relevância. Defenderemos que este postulado é um bom candidato para substituir a recuperação e o compararemos com a recuperação ótima. Nossa análise se apoiará em três critérios.

1. *equivalência com a construção partial meet:* o teorema de representação 6.8 mostra que a construção partial meet é equivalente aos postulados AGM para lógicas que satisfazem as suposições AGM. Em que lógicas essa equivalência vale ao trocarmos a recuperação pela relevância? Qual conjunto de postulados melhor caracteriza a contração partial meet?

2. *compatibilidade:* os postulados AGM são compatíveis com as lógicas decomponíveis. Que propriedade(s) uma lógica deve satisfazer para ser compatível com os postulados da contração-relevante? Ela é compatível com quais lógicas apresentadas no capítulo 5.4?

3. *racionalidade AGM:* em [Han91] foi provado que a recuperação é equivalente a relevância (na presença dos outros postulados AGM) para lógicas que satisfazem as suposições AGM. Esse resultado vale para qualquer lógica? Ele vale para qualquer lógica decomponível?

Nossa abordagem consiste em verificar quais das propriedades lógicas apresentadas no capítulo 5 (distributividade, fecho por negação, decomponibilidade e compacidade) garantem a compatibilidade, a racionalidade e a equivalência com partial meet.

6.1.1 Relevância e Racionalidade AGM

Hansson provou que, para lógicas que satisfazem as suposições AGM, a relevância e a recuperação são equivalentes na presença dos demais postulados AGM [Han91]. Nessa seção mostraremos que esse resultado não vale em geral para lógicas não clássicas.

Consideraremos quatro propriedades lógicas nesta seção: distributividade, fecho por negação, decomponibilidade e compacidade. Nossa abordagem consistirá em mostrar quais subconjuntos destas propriedades lógicas garantem que a relevância seja equivalente a recuperação.

Primeiramente definiremos melhor o que queremos dizer por equivalência entre recuperação e relevância.

Definição 6.2. *Dizemos que em uma lógica $\langle \mathscr{L}, Cn\rangle$ a relevância implica a recuperação (na presença dos demais postulados AGM) sse para todo conjunto de crenças K temos que a operação $-$ para K satisfaz a recuperação se essa operação satisfizer os demais postulados AGM e a relevância.*

Analogamente dizemos que a recuperação implica a relevância sse para todo conjunto de crenças K a operação $-$ para K satisfaz os postulados AGM então $-$ satisfaz a relevância.

Dizemos que relevância e recuperação são equivalentes (na presença dos demais postulados) em $\langle \mathscr{L}, Cn\rangle$ se a recuperação implica a relevância e a relevância implica a recuperação.

Com essa definição mais precisa podemos mostrar que relevância e recuperação não são equivalentes em qualquer lógica $\langle \mathscr{L}, Cn\rangle$. O exemplo 5.1 mostra que a distributividade junto da compacidade não são suficientes para garantir a relevância implica a recuperação:

Observação 6.3. *O exemplo 5.1 mostra uma lógica $\langle \mathscr{L}, Cn\rangle$ finita e distributiva. Considere o conjunto de crenças $\mathscr{L} = Cn(\{a, b\}) = Cn(b)$ e a seguinte operação $-$ para $\mathscr{L}$:*

$$\begin{aligned} \mathscr{L} - \mathscr{L} &= \{a\} \\ \mathscr{L} - \{a\} &= \emptyset \\ \mathscr{L} - \{b\} &= \{a\} \\ \mathscr{L} - \emptyset &= \mathscr{L} \end{aligned}$$

Essa operação satisfaz a relevância, mas não satisfaz a recuperação. Pois, $\mathscr{L} - \{a\} + \{a\} = \{a\} \neq \mathscr{L}$.

O postulado da recuperação ótima [FPA06] pode ser considerado uma generalização da recuperação por ser equivalente a recuperação AGM em toda lógica AGM compatível. Ou seja, para toda lógica em que a contração AGM é bem definida, a recuperação ótima é equivalente a recuperação AGM na presença dos demais postulados.

A relevância não possui essa mesma característica. A lógica do exemplo 5.16 é decomponível, mas nela a recuperação não implica a relevância assim como a relevância não implica a recuperação.

Observação 6.4. *O exemplo 5.16 mostra uma lógica $\langle \mathscr{L}, Cn\rangle$ finita e decomponível, portanto, fechada por negação. Considere o conjunto de crenças $\mathscr{L}$ e a seguinte contração $-$ para $\mathscr{L}$:*

$$\begin{aligned} \mathscr{L} - \mathscr{L} &= \{c\} \\ \mathscr{L} - \{a, c\} &= \{c\} \\ \mathscr{L} - \{b, c\} &= \{c\} \\ \mathscr{L} - \{a, b\} &= \{a\} \\ \mathscr{L} - \{a\} &= \{b\} \\ \mathscr{L} - \{b\} &= \{a\} \\ \mathscr{L} - \{c\} &= \{a, b\} \\ \mathscr{L} - \emptyset &= \mathscr{L} \end{aligned}$$

Essa contração satisfaz a relevância, mas não satisfaz a recuperação pois $\mathscr{L} - \{a, b\} + \{a, b\} = \{a, b\} \neq \mathscr{L}$.

Agora considere a seguinte contração em $\mathscr{L}$*:*

$$\begin{aligned}
\mathscr{L} - \mathscr{L} &= \{c\} \\
\mathscr{L} - \{a, c\} &= \{c\} \\
\mathscr{L} - \{b, c\} &= \{c\} \\
\mathscr{L} - \{a, b\} &= \{c\} \\
\mathscr{L} - \{a\} &= \{c\} \\
\mathscr{L} - \{b\} &= \{c\} \\
\mathscr{L} - \{c\} &= \{a, b\} \\
\mathscr{L} - \emptyset &= \mathscr{L}
\end{aligned}$$

Essa contração satisfaz a recuperação, mas não satisfaz a relevância. Porque $b \notin \mathscr{L} \setminus \mathscr{L} - \{c\}$ *e não existe* K' *tal que* $\mathscr{L} - \{c\} \subseteq K' \subseteq \mathscr{L}$, $c \notin Cn(K')$ *e* $c \in Cn(K' \cup \{b\})$.

A proposição a seguir mostra que para lógicas distributiva *e* fechadas por negação esses postulados são equivalentes na presença dos outros postulados AGM. Ou seja, a equivalência entre a relevância e a recuperação vale para qualquer lógica booleana:

Proposição 6.5. *Em lógicas booleanas, os postulados AGM são equivalentes aos postulados da contração-relevante.*

Demonstração. Veja o apêndice B.2 □

Portanto, em lógicas finitas, a distributividade e a decomponibilidade implicam que a relevância e a recuperação são equivalentes:

Corolário 6.6. *Para lógicas finitas, distributivas e decomponíveis, a relevância e a recuperação são equivalentes na presença dos outros postulados AGM.*

A tabela a seguir resume os resultados apresentados nessa seção e os compara com os resultados de [FPA06]. A tabela mostra quais propriedades lógicas garantem que os postulados AGM são equivalentes aos postulados da contração-relevante e aos postulados AGM com recuperação ótima. Por exemplo, a primeira linha da tabela indica que a distributividade não garante que a contração AGM seja equivalente a contração-relevante, nem que a contração AGM seja equivalente a contração com recuperação ótima.

	relevância	recuperação ótima
distributiva	não (obs. 6.3)	não [FPA06]
negação	não (obs. 6.4)	?
booleana	sim (prop. 6.5)	sim [FPA06]
decomponível	não (obs. 6.4)	sim [FPA06]
decomponível + finita + distributiva	sim (theo. 6.6)	sim [FPA06]

Tabela 6.1: Racionalidade AGM

6.1.2 Relevância e Partial Meet

Na Seção 3.2 vimos que para as lógicas que satisfazem as suposições AGM a contração partial meet é equivalente aos postulados AGM [AGM85]. Esse resultado central na teoria AGM foi

chamado de teorema de representação da contração partial meet. O teorema da representação nos mostra que os postulados AGM caracterizam precisamente a construção partial meet.

No começo deste capítulo argumentamos que os postulados AGM não são aplicáveis a diversas lógicas que nos interessam. Sugerimos então substituir o postulado da recuperação pela relevância ou pela recuperação ótima.

O resultado principal dessa seção mostra que os postulados da contração-relevante caracterizam a contração partial meet em uma classe maior de lógicas do que os postulados AGM. Mais precisamente, o teorema de representação que relaciona a contração-relevante com a contração partial meet vale para qualquer lógica compacta.

Antes de mostrar esse resultado vamos generalizar a construção do conjunto resíduo:

Definição 6.7 (conjunto resíduo)**.** *Seja K um conjunto de crenças e A um conjunto de sentenças. O* conjunto resíduo $K \perp A$ *é o conjunto tal que $X \in K \perp A$ sse:*

1. $X \subseteq K$ *(subconjunto de K)*
2. $A \nsubseteq Cn(X)$ *(que não implica A)*
3. *se $X \subset X' \subseteq K$ então $A \subseteq Cn(X')$ (maximal)*

Agora podemos enunciar o teorema de representação:

Teorema de Representação 6.8. *Seja $\langle \mathscr{L}, Cn\rangle$ uma lógica compacta e seja A finitamente representável, então $K - A$ satisfaz a relevância e os demais postulados AGM sse $K - A = \bigcap \gamma(K \perp A)$ para alguma função de seleção γ.*

Demonstração. Veja o apêndice B.2. □

No restante dessa seção mostraremos lógicas em que partial meet não é equivalente aos postulados AGM.

Os exemplos 5.1 e 5.16 mostram lógicas em que os postulados AGM não são equivalentes a contração partial meet. Essas lógicas finitas são distributiva e decomponível respectivamente. Logo, mesmo em lógicas finitas, nem distributividade, nem decomponibilidade garantem que a contração partial meet seja caracterizada pelos postulados AGM (mesmo com a recuperação ótima).

Observação 6.9. *O exemplo 5.1 mostra uma lógica distributiva $\langle \mathscr{L}, Cn\rangle$ tal que a contração partial meet não é equivalente aos postulados AGM (mesmo trocando a recuperação pela recuperação-ótima). Para verificar esse fato pegue $K = \mathscr{L}$ e $A = \{a\}$.*

Observação 6.10. *O exemplo 5.16 mostra uma lógica decomponível $\langle \mathscr{L}, Cn\rangle$ tal que a contração partial meet não é equivalente aos postulados AGM (mesmo trocando a recuperação pela recuperação-ótima). Para verificar esse fato pegue $K = \mathscr{L}$ e $A = \{a, b\}$.*

Note que o teorema de representação 6.8 não vale para qualquer lógica (não compacta). A lógica do exemplo 5.17 é booleana, não é compacta e nela os postulados da contração-relevante não caracterizam a contração partial meet.

Observação 6.11. *O exemplo 5.17 mostra uma lógica booleana $\langle \mathscr{L}, Cn\rangle$ tal que a contração partial meet não é equivalente aos postulados da contração-relevante. Para verificar esse fato pegue $K = \mathscr{L}$ e $A = \{a\}$.*

Concluímos que a contração partial meet é melhor caracterizada pelos postulados da contração-relevante do que pelos postulados AGM ou pelos os postulados AGM com recuperação-ótima. Porém, essa equivalência entre contração partial meet e contração-relevante não vale em geral para lógicas não compactas.

Assim, por exemplo, na lógica de Horn, que é compacta, a contração partial meet é caracterizada pelos postulados da contração-relevante, mas não é caracterizada pelos postulados da contração AGM por não ser decomponível.

A tabela a seguir resume os resultados dessa seção.

	AGM	relevância	rec. ótima
decomponível	não (obs. 6.10 e 6.11)	não (obs. 6.11)	não [FPA06]
finita	não (obs. 6.10 e 6.9)	sim (rep. theo. 6.8)	não [FPA06]
compacta	não (obs. 6.10 e 6.9)	sim (rep. theo. 6.8)	não [FPA06]
booleana + compacta	sim [AGM85]	sim (teo. rep. 6.8)	sim [FPA06]

Tabela 6.2: Partial meet

6.1.3 Relevância e Compatibilidade

O último aspecto da contração-relevante que iremos abordar é sua compatibilidade com lógicas não clássicas. Na Seção 5.1 mostramos que nem toda lógica tarskiana é compatível com os postulados AGM. Mais precisamente, vimos que uma lógica é compatível com AGM se e somente se for decomponível [FPA05].

Primeiramente vamos generalizar esse conceito de compatibilidade:

Definição 6.12 (compatibilidade)**.** *Uma lógica $\langle \mathscr{L}, Cn \rangle$ e um conjunto de postulados para uma operação $-$ para K são* compatíveis *entre si sse para todo K existe pelo menos uma operação $-$ para K que satisfaz todos os postulados.*

Flouris [Flo06] mostrou que qualquer lógica tarskiana é compatível com os postulados withdrawal, mas nem toda lógica tarskiana é compatível com os postulados AGM.

Proposição 6.13. *[Flo06] Toda lógica tarskiana é compatível como os postulados withdrawal.*

Este resultado junto do teorema 5.4 nos mostra que o problema da compatibilidade AGM está na relação entre o postulado da recuperação e os demais postulados. No começo deste capítulo sugerimos substituir a recuperação pela relevância e definimos assim a contração-relevante. Desejamos que, diferentemente da contração AGM, a contração-relevante seja compatível com as lógicas apresentadas no capítulo 5.

Nessa seção mostraremos quais propriedades lógicas garantem a compatibilidade com os postulados da contração-relevante.

Como mostramos na seção anterior, em qualquer lógica compacta a contração partial meet é caracterizada pelos postulados da contração-relevante. Um corolário direto desse resultado é a compatibilidade entre as lógicas compactas e os postulados da contração-relevante:

Corolário 6.14. *Toda lógica compacta $\langle \mathscr{L}, Cn \rangle$ é compatível com os postulados da contração-relevante.*

Esse resultado vale pois para todo $K, A \in 2^{\mathscr{L}}$ podemos fazer $K - A = \bigcap \gamma(K \perp A)$ para qualquer γ e teremos uma contração que satisfaz os postulados da contração-relevante.

Apesar de bastante geral, esse resultado não pode ser generalizado para qualquer lógica (não compacta):

Observação 6.15. *O exemplo 5.17 mostra uma lógica booleana $\langle \mathscr{L}, Cn \rangle$ que não é compatível com os postulados da contração-relevante.*

Por fim, cabe mencionar que a compatibilidade entre uma lógica $\langle \mathscr{L}, Cn \rangle$ e os postulados da recuperação-ótima vale para qualquer lógica decomponível *ou* finita [FPA06].

No capítulo 5 mostramos diversas lógicas não decomponíveis e compactas: lógicas de descrição com hierarquia de papéis e sem construtores entre papéis, lógicas de descrição com ABox e sem nominais, a lógica de Horn e a lógica intuicionista. Todas essas lógicas são compatíveis com a contração-relevante, mas não são compatíveis com a contração AGM. Para essas lógicas não precisamos aceitar a recuperação como uma "propriedade emergente", pois nelas a recuperação e a relevância (junto dos outros postulados AGM) definem operações distintas. Podemos nesses casos escolher se damos preferência a recuperabilidade definida pela recuperação ou a minimalidade definida pela relevância.

A tabela a seguir resume os resultados de compatibilidade apresentados nessa seção:

	AGM	relevância	rec. ótima	withdrawal
decomponível	sim [FPA05]	não (obs. 6.15)	sim [FPA06]	sim [Flo06]
finita	não (ex. 5.1)	sim (cor. 6.14)	sim [FPA06]	sim [Flo06]
compacta	não (ex. 5.1)	sim (cor. 6.14)	?	sim [Flo06]

Tabela 6.3: Compatibilidade

6.2 Conclusão

Para resolver o problema da incompatibilidade AGM com as lógicas do Capítulo 5 propusemos substituir a recuperação pela relevância nos postulado AGM para contração. A relevância é um postulado proposto por Hansson que captura mais diretamente a noção de minimalidade. Estudamos essa operação que chamamos de contração-relevante em três aspectos.

Primeiro mostramos que em lógicas booleanas a contração-relevante é equivalente a contração AGM. Porém, essa equivalência não vale para qualquer lógica. Em diversas lógicas a relevância e a recuperação são postulados distintos. Nessas lógicas podemos optar se preferimos priorizar a recuperabilidade garantida pela recuperação ou a minimalidade garantida pela relevância. Nesta primeira parte respondemos a questão que nos fizemos sobre a racionalidade da relevância.

Em seguida mostramos que os postulados da contração-relevantecaracterizam melhor a contração partial meet. A equivalência entre ospostulados da contração-relevante e a contração partial meet vale paraqualquer lógica compacta enquanto que a equivalência entre ospostulados da contração AGM com a contração partial meet vale apenasem lógicas que satisfazem as suposições AGM. Desta forma, respondemos a primeira pergunta que nos fizemos neste capítulo.

Por fim, respondemos a segunda pergunta mostrando que a contração-relevante é compatível com qualquer lógica compacta. Ou seja, ela é compatível com a maioria das lógicas do Capítulo 5 (veja a tabela 5.1).

Esses três aspectos da contração-relevante são resumidos nas tabelas 6.1.1, 6.1.2 e 6.1.3 respectivamente.

Os resultados acima indicam que em lógicas não clássicas o postulado da relevãncia é mais adequado do que o postulado da recuperação. Apesar de menos elegante, a relevãncia define a minimalidade de forma direta, é compatível com uma classe maior de lógicas e caracteriza a contração partial meet para uma classe maior de lógicas.

	equivalência com partial meet		compatibilidade				racionalidade AGM	
	AGM	relevância	AGM	relevância	withdrawal	rec. ótima	relevância	rec. ótima
compacta	não (obs. 6.9)	sim (theo. 6.8)	não (ex. 5.1)	sim (cor. 6.14)	sim [Flo06]	?	não (obs. 6.3)	?
finita	não (obs. 6.9)	sim (theo. 6.8)	não (ex. 5.1)	sim (cor. 6.14)	sim [Flo06]	sim [FPA06]	não (obs. 6.3)	?
negação	não (obs. 6.10)	não (obs. 6.11)	?	não (obs. 6.15)	sim [Flo06]	?	não (obs. 6.4)	?
distributiva	não (obs. 6.9)	não (obs. 6.11)	não (ex. 5.1)	não (obs. 6.15)	sim [Flo06]	?	não (obs. 6.3)	?
booleana	não (obs. 6.11)	não (obs. 6.11)	sim [FPA05]	não (obs. 6.15)	sim [Flo06]	sim [FPA06]	sim (theo. 6.5)	sim [FPA06]
booleana + compacta	sim [AGM85]	sim (theo. 6.8)	sim [FPA05]	sim (cor. 6.14)	sim [Flo06]	sim [FPA06]	sim (theo. 6.5)	sim [FPA06]
decomponível	não (obs. 6.10)	não (obs. 6.11)	sim [FPA05]	não (obs. 6.15)	sim [Flo06]	sim [FPA06]	não (obs. 6.4)	sim [FPA06]

Tabela 6.4: Resumo dos resultados da Seção 6.1

Capítulo 7

Revisão AGM em Lógicas sem Negação

Na Seção 3.2.3 mostramos como definir uma revisão AGM a partir de uma contração AGM usando a identidade de Levi ($K * \alpha = K - \neg\alpha + \alpha$). A identidade de Levi faz uso direto da negação de uma sentença α. Porém, como discutido na Seção 5, diversas lógicas interessantes não são fechadas por negação.

Neste capítulo mostraremos uma construção para a revisão AGM que não depende da negação. Como no capítulo anterior definiremos a operação entre um conjunto de crenças K e um conjunto de sentenças A. A construção que apresentaremos é caracterizada pelos postulados AGM em lógicas que satisfazem as suposições AGM. Como estamos interessados em outras lógicas, mostraremos um conjunto de postulados que caracteriza a construção para uma classe maior de lógicas: a classe das lógicas distributivas e compactas.

7.1 Construção

A construção que buscamos, além de não usar a negação, deve satisfazer os postulados AGM para revisão.

A estratégia usada para construir tal revisão se assemelha à estratégia usada pela identidade de Levi. Primeiro removemos de K algumas sentenças de forma que, ao adicionar A, o novo conjunto de crenças $K * A$ seja consistente. Faremos isso definindo o conjunto $K \downarrow A$:

Definição 7.1 ([Del08])**.** *Os* subconjuntos maximais de K que junto de A são consistentes*, formam o conjunto $K \downarrow A$ que é definido como $X \in K \downarrow A$ sse:*

- $X \subseteq K$.
- $X \cup A$ *é consistente.*
- *Se* $X \subset X' \subseteq K$ *então* $X' \cup A$ *é inconsistente.*

Uma função de seleção é definida de forma usual como:

Definição 7.2 (função de seleção)**.** *Uma* função de seleção *para K é uma função γ tal que:*

- $\emptyset \neq \gamma(K \downarrow A) \subseteq K \downarrow A$ *se* $K \downarrow A \neq \emptyset$.
- $\gamma(K \downarrow A) = \{K\}$ *caso contrário.*

A *revisão AGM sem negação* é definida da seguinte forma:

$$K *_\gamma A = \bigcap \gamma(K \downarrow A) + A$$

Primeiro tomamos a intersecção de alguns (pelo menos um) elementos de $K \downarrow A$ e em seguida expandimos por A.

O primeiro resultado dessa seção mostra que a construção satisfaz todos os postulados AGM para revisão:

Proposição 7.3. *Considerando a lógica subjacente tarskiana e compacta, temos que $K *_\gamma A$ satisfaz todos os postulados AGM para revisão.*

Demonstração. Veja o apêndice B.3 □

7.2 Postulados e Teoremas de Representação

Na seção anterior mostramos uma construção para revisão que satisfaz todos os postulados AGM. Temos dois objetivos nessa seção. O primeiro é mostrar um conjunto de postulados que caracteriza a construção acima para qualquer lógica compacta e distributiva. O segundo é mostrar que a construção acima é caracterizada pelos postulados AGM caso a lógica subjacente satisfaça as suposições AGM.

No fim da seção anterior mostrar que a construção apresentada satisfaz todos os postulados AGM para revisão caso a lógica subjacente seja tarskiana e compacta. Os postulados AGM não são suficientes porém para caracterizar essa construção para essa classe de lógicas.

Consideraremos dois seguintes postulados importados da área de revisão em bases de crenças: uniformidade e relevância.

(uniformidade [Han93]) Se para todo $K' \subseteq K$, $K' \cup A$ é inconsistente sse $K' \cup B$ é inconsistente, então $K \cap K * A = K \cap K * B$.

Esse postulado garante que se duas entradas α e β são inconsistentes com os mesmos subconjuntos de K, então as sentenças removidas em $K * \alpha$ devem ser as mesmas que em $K * \beta$.

(relevância) Se $\beta \in K \setminus K * \alpha$ então existe K' tal que $K \cap K * \alpha \subseteq K' \subseteq K$ e $K' \cup \{\alpha\}$ é consistente, mas $K' \cup \{\alpha, \beta\}$ não.

A relevância garante que só é permitido remover sentenças de K que estejam de algum forma envolvidas em derivar inconsistências. A relevância evita perda desnecessária de informação.

A construção apresentada na seção anterior satisfaz tanto a relevância quanto a uniformidade em qualquer lógica tarskiana:

Proposição 7.4. *$K *_\gamma A = \bigcap \gamma(K \downarrow A) + A$ satisfaz a uniformidade e a relevância.*

Além disso, na presença desses dois postulados a vacuidade e a extensionalidade são redundantes:

Proposição 7.5. *Considere que a lógica subjacente é distributiva e tarskiana. Se um operador $*$ satisfaz:*

1. *sucesso, inclusão e relevância então $*$ satisfaz a vacuidade.*
2. *sucesso, inclusão e uniformidade então $*$ satisfaz a extensionalidade.*

Demonstração. Veja o apêndice B.3 □

Com esses postulados somos capazes de caracterizar a construção acima para lógicas que não são fechadas por negação. Essa caracterização vale para lógicas tarskianas, distributivas e compactas:

Teorema de Representação 7.6 (Revisão AGM sem negação)**.** *Para qualquer lógica tarskiana, compacta, distributiva que satisfaz a explosão fraca, o operador de revisão '$*$' é uma revisão AGM sem negação ($K *_\gamma A = \bigcap \gamma(K \downarrow A) + A$) para alguma função de seleção γ sse $*$ satisfaz fecho, sucesso, inclusão, consistência, relevância e uniformidade.*

No Capítulo 5 mostramos algumas lógicas que não são fechadas por negação, mas são distributivas e compactas: diversas lógicas de descrição como $\mathcal{ALC}$ (sem ABox) e $\mathcal{ALCO}$ e a lógica intuicionista. O teorema de representação acima se aplica a essas lógicas.

Mencionamos no inicio dessa seção que para as lógicas que satisfazem as suposições AGM, a revisão AGM sem negação é caracterizada pelos postulados AGM. Mostraremos isso provando que nessas lógicas os postulados AGM são equivalentes aos postulados usados no teorema de representação 7.6:

Proposição 7.7. *Assuma que a lógica subjacente satisfaça as suposições AGM. Um operador $*$ satisfaz os postulados AGM sse $*$ satisfaz: fecho, sucesso, inclusão, consistência, relevância e uniformidade.*

Demonstração. Veja o apêndice B.3 □

7.3 Conclusão

No capítulo 5 mostramos que muitas lógicas de interesse não são fechadas por negação. Por exemplo: grande parte das LDs, a lógica de Horn etc. Nessas lógicas não é possível aplicar diretamente a identidade de Levi. Neste capítulo mostramos como construir uma revisão em conjuntos de crenças que não dependa da negação.

A construção apresentada é caracterizada pelos postulados AGM caso a lógica subjacente satisfaça as suposições AGM. Além disso, mostramos um conjunto de postulados que caracteriza essa construção em uma classe mais ampla de lógicas que inclui várias das lógicas que nos interessam: as lógicas distributivas e compactas.

Capítulo 8

Bases de Crenças em Lógicas sem Negação

No capítulo 3 mostramos diversas construções para a revisão em bases de crenças. Todas essas construções assumem que a lógica subjacente seja monotônica, compacta e fechada por negação que satisfaz α não contravenção local. Nesse capítulo apresentamos construções para a revisão em bases de crenças que não dependem da negação. Essas construções podem ser aplicadas a qualquer lógica compacta e monotônica.

A importância dessa generalização está na dificuldade de, em algumas lógicas, explicitar qual seria a negação de determinada sentença. Por exemplo, mostramos na Seção 5.4.5 que a sentença $\top \sqsubseteq A$ na lógica $\mathcal{ALC}$ possui uma infinidade de conjuntos que satisfazem α não contravenção local. Qualquer um desses conjuntos poderia ser usada nas construções da Seção 3.3. Para evitar uma escolha arbitrária, definimos a revisão sem negação.

Como no capítulo anterior, as revisões serão definidas entre conjuntos de sentenças ($B * A$). Além disso, como explicamos no capítulo 4, assumiremos que existe um conjunto previamente definido de sentenças indesejadas Ω. Chamaremos um conjunto de sentenças A de *inconsistente* sse $Cn(A) \cap \Omega \neq \emptyset$.

Neste trabalho não nos preocupamos com a *contração* em bases de crença uma vez que os resultados da Seção 3.3 são suficientemente gerais para contemplar todas as lógicas que nos interessam (veja a tabela 5.1).

Neste capítulo definiremos seis construções para revisão em bases de crenças. Na Seção 8.1.1 mostraremos duas generalizações para a revisão externa kernel, uma que privilegia o sucesso (revisão externa kernel sem negação com sucesso forte) e outra que privilegia a consistência (revisão externa kernel sem negação com consistência forte). Na seção seguinte mostraremos duas generalizações para a revisão externa partial meet (revisão externa partial meet sem negação com sucesso forte e com consistência forte). Nas seções 8.2.1 e 8.2.2 generalizaremos a revisão interna partial meet (revisão interna partial meet sem negação) e a revisão interna kernel (revisão interna kernel sem negação) respectivamente. O esquema da figura 8.1 ilustra a divisão de seções desse capítulo. Para cada uma dessas revisões definiremos a construção, o conjunto de postulados que a caracteriza e o teorema de representação que prova essa caracterização.

8.1 Revisão Externa sem Negação

A construção de uma revisão externa [Han93] consiste de uma expansão seguida de uma contração:

$$B + A - \neg A$$

Essa forma de construir a revisão, porém, depende da existência da negação de A. Como explicamos na introdução desse capítulo, nosso objetivo é evitar o uso da negação. Para isso,

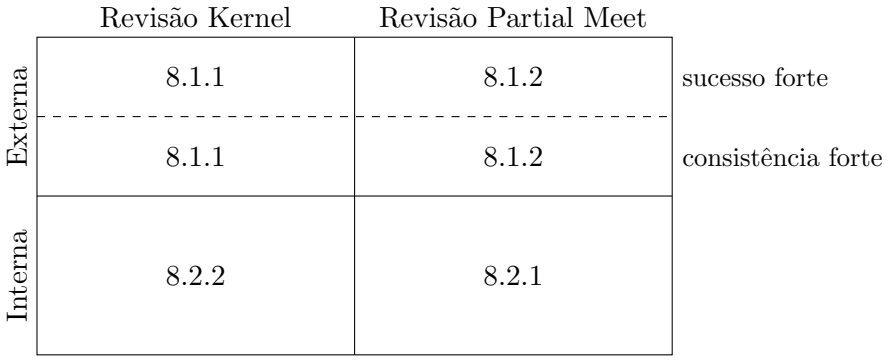


Figura 8.1: Seções

usaremos uma estratégia inspirada na idéia de semi-revisão [Han97] (Seção 3.3.3).

Como na semi-revisão, a idéia é inserir A na base B e em seguida *remover as inconsistências*. Em uma semi-revisão, porém, esse processo pode levar a remoção de A. Por isso a semi-revisão não satisfaz o sucesso. Para garantir o *sucesso* evitamos excluir elementos de A ao remover as inconsistências. Em outras palavras, removemos as inconsistências de $B + A$ *protegendo* a entrada A.

Quando A é inconsistente é impossível não excluir nenhum de seus elementos no processo de remoção das inconsistências. Esse é o único caso em que é impossível proteger A. Nesse caso temos que escolher qual postulado de racionalidade vamos priorizar: sucesso ou consistência. Podemos optar por garantir o sucesso a todo custo (sucesso forte) e uma versão fraca da consistência. Alternativamente podemos garantir a consistência forte e apenas uma versão fraca do sucesso (sucesso fraco). A primeira opção é a que mais se aproxima da tradição AGM.

(sucesso forte) $A \subseteq B * A$.

(sucesso fraco) Se A é consistente então $A \subseteq B * A$.

(consistência forte) $B * A$ é consistente.

(consistência fraca) Se A é consistente então $B * A$ é consistente.

Na Seção 3.3 mostramos duas maneiras de remover as inconsistências de uma base de crenças: contração kernel e contração partial meet. Como cada um desses métodos pode ser combinado com sucesso forte ou consistência forte, definimos quatro formas de revisão externa: revisão externa kernel com sucesso forte (REKS), revisão externa kernel com consistência forte (REKC), revisão externa partial meet com sucesso forte (REPMS) e revisão externa partial meet com consistência forte (REPMC).

O postulado de minimalidade relacionado à revisão kernel é o core-retainment e o postulado relacionado à revisão partial meet é a relevância. Ambos os postulados usam a negação de sentenças. Vamos adaptá-los de forma a não mencionar a negação:

(core-retainment) Se $\beta \in B * A \setminus B$ então existe $B' \subseteq B \cup A$ tal que B' é consistente, mas $B' \cup \{\beta\}$ não é.

(relevância) Se $\beta \in B * A \setminus B$ então existe B' tal que $B * A \subseteq B' \subseteq B \cup A$ tal que B' é consistente, mas $B' \cup \{\beta\}$ não é.

No caso em que A possui um único elemento e a lógica subjacente é fechada por negação, temos que esses postulados são equivalentes aos postulados da Seção 3.3.2. Além disso, na presença da inclusão (em bases) e do sucesso forte, a relevância apresentada aqui é equivalente a relevância usada na revisão AGM sem negação apresentada na Seção 7.

Nesse capítulo sempre que usarmos o termo *relevância* nos referimos ao postulado definido acima. Nos casos em que vale o sucesso forte e a inclusão usaremos qualquer das duas definições indistintamente, dado que elas são equivalentes nesse caso[1]

Proposição 8.1. *Na presença do sucesso forte e da inclusão a relevância apresentada nessa seção é equivalente à relevância apresentada na Seção 7.*

Demonstração. Veja o apêndice B.4.1. □

Os demais postulados precisam ser generalizados para aceitar conjuntos de sentenças como entrada. A generalização desses postulados é direta:

(inclusão) $B * A \subseteq B + A$

(pré-expansão) $B * A = (B + A) * A$

Diferente das revisões apresentadas no capítulo 3, as revisões apresentadas nesse capítulo não satisfazem a uniformidade fraca. Isso acontece porque não protegemos entradas equivalentes uniformemente. Ou seja, ao protegermos a entrada violamos a independência sintática.

Nas próximas seções definimos uma construção para cada uma das revisões (REKS, REKC, REPM e REPMC), mostramos que cada uma delas é caracterizada por determinado conjunto de postulados e provamos os respectivos teoremas de representação.

8.1.1 Revisão Externa Kernel sem Negação

A *revisão externa kernel sem negação* $(B * A)$ é definida de forma similar a uma semi-revisão kernel. Primeiro expandimos a base B por A (em símbolos $B + A$) e em seguida aplicamos uma contração kernel que protege a entrada[2]:

$$B + A \setminus \sigma(A, B + A \perp\!\!\!\perp \Omega)$$

A contração kernel, nesse caso, consiste em remover pelo menos um elemento de cada conjunto minimal inconsistente. Como generalizamos o sentido de inconsistência, temos que generalizar a definição de conjunto kernel. A generalização é feita de forma natural:

Definição 8.2 (Kernel $(B \perp\!\!\!\perp \Omega)$)**.** $X \in B \perp\!\!\!\perp \Omega$ *sse:*

1. $B' \subseteq B$
2. B' *é inconsistente*
3. *Se* $B'' \subset B'$ *então* B'' *é consistente.*

Chamamos X *de* subconjunto minimal inconsistente de B.

> **Exemplo 8.3:** Considere a modelagem para o exemplo 1.1 apresentada na introdução. Assumiremos que Ω é um conjunto de sentenças que não possui modelos. Juntando todas as sentenças do exemplo temos o seguinte conjunto inconsistente B:
>
> $Saudável \sqsubseteq Alimento \sqcap \neg\exists rico_em.Gordura$
>
> $Alimento \sqcap \exists rico_em.Gordura(margarina)$
>
> $Alimento \sqcap \exists rico_em.Gordura(manteiga)$
>
> $Trans \sqsubseteq Gordura$

[1] Tradicionalmente usa-se nomes iguais para postulados análogos em revisão em conjuntos de crenças e bases de crenças. Sempre que não for claro no contexto, explicitaremos se o postulado em questão é o de bases ou o de conjuntos de crenças.

[2] Para manter a notação limpa assumimos que o operador $+$ precede tanto o operador $\perp\!\!\!\perp$ quanto o operador $\perp$ e.g. $B + A \perp\!\!\!\perp \Omega = (B + A) \perp\!\!\!\perp \Omega$.

$Saudável \sqsubseteq Alimento \sqcap \neg \exists rico_em.Trans$

$\exists rico_em.Trans(margarina)$

$Saudável(manteiga)$

$Saudável(margarina)$

O conjunto $B \perp\!\!\!\perp \Omega$ é formado pelos seguintes elementos:

$\{Saudável(margarina),$
$\quad \exists rico_em.Trans(margarina),$
$\quad Saudável \sqsubseteq Alimento \sqcap \neg \exists rico_em.Trans\}$

$\{Saudável(margarina),$
$\quad \exists rico_em.Trans(margarina),$
$\quad Trans \sqsubseteq Gordura,$
$\quad Saudável \sqsubseteq Alimento \sqcap \neg \exists rico_em.Gordura\}$

$\{Saudável(margarina),$
$\quad \exists rico_em.Gordura(margarina),$
$\quad Saudável \sqsubseteq Alimento \sqcap \neg \exists rico_em.Gordura\}$

$\{Saudável(manteiga),$
$\quad \exists rico_em.Gordura(manteiga),$
$\quad Saudável \sqsubseteq Alimento \sqcap \neg \exists rico_em.Gordura\}$

Uma função de incisão escolhe pelo menos um elemento de cada conjunto minimal inconsistente. Para garantir o sucesso (forte ou fraco) definiremos a função de incisão que protege a entrada. Definiremos dois tipos de função de incisão: a primeira protege qualquer entrada A e garante o sucesso forte; a segunda protege a entrada A apenas se A for consistente e, assim, garante a consistência forte.

Revisão Externa Kernel sem Negação com Sucesso Forte (REKS)

Primeiramente definiremos a *função de incisão que protege qualquer entrada.* Esta função recebe dois valores. O primeiro é a entrada que será protegida A e o segundo é o conjunto kernel $B \perp\!\!\!\perp \Omega$.

Como a função de incisão da definição 3.22, essa função escolhe pelo menos um elemento de cada conjunto minimal inconsistente (caso A seja consistente). Diferente da função da definição 3.22, essa função nunca escolhe elementos de A. Os elementos de A serão os únicos elemento de um conjunto minimal inconsistente sse A for inconsistente. Apenas nesse caso não garantimos que será escolhido pelo menos um elemento de cada conjunto minimal inconsistente.

Definição 8.4 (Função de incisão (σ) que protege a entrada (A))**.** *Uma* função de incisão σ que protege a entrada A *é definida como:*

1. $\sigma(A, B \perp\!\!\!\perp \Omega) \subseteq \bigcup(B \perp\!\!\!\perp \Omega)$.
2. *Se* A *é inconsistente e* $\emptyset \neq X \in B \perp\!\!\!\perp \Omega$, *então* $X \cap \sigma(A, B \perp\!\!\!\perp \Omega) \neq \emptyset$.
3. $A \cap \sigma(A, B \perp\!\!\!\perp \Omega) = \emptyset$.

Exemplo 8.5: Considere uma base de crenças B formada pelos seguintes elementos:

$Saudável \sqsubseteq Alimento \sqcap \neg \exists rico_em.Gordura$

$Alimento \sqcap \exists rico_em.Gordura(margarina)$

$Alimento \sqcap \exists rico_em.Gordura(manteiga)$

$Trans \sqsubseteq Gordura$
$Saudável \sqsubseteq Alimento \sqcap \neg \exists rico_em.Trans$
$\exists rico_em.Trans(margarina)$

Assumindo que Ω é inconsistente e que a entrada A seja:

$Saudável(manteiga)$
$Saudável(margarina)$

Temos que $B + A \perp\!\!\!\perp \Omega$ é o conjunto apresentado no exemplo 8.3.

Uma função de incisão σ que protege a entrada não pode escolher $Saudável(manteiga)$ nem $Saudável(margarina)$ de $B + A \perp\!\!\!\perp \Omega$.

A *revisão externa kernel sem negação e com sucesso forte* é definida usando a função de incisão que protege a entrada σ como:

$$B * A = B + A \setminus \sigma(A, B + A \perp\!\!\!\perp \Omega)$$

Essa revisão é caracterizada pelos seguintes postulados: inclusão, pré-expansão, core-retainment, sucesso forte e consistência fraca. O core-retainment está relacionado com o fato de usarmos a contração kernel para remover as inconsistências. O sucesso forte e a consistência fraca estão relacionados com o uso da função de incisão que protege qualquer entrada.

Teorema de Representação 8.6 (REKS)**.** *Uma* revisão externa kernel sem negação e com sucesso forte *(REKS) $B * A$ satisfaz sucesso forte, consistência fraca, inclusão, core-retainment e pré-expansão sse existe uma função de incisão σ que protege a entrada tal que $B * A = B + A \setminus (A, B + A \perp\!\!\!\perp \Omega)$.*

Demonstração. Veja o apêndice B.4.2 □

Revisão Externa Kernel sem Negação com Consistência Forte (REKC)

A construção apresentada acima prioriza o sucesso em detrimento da consistência. A construção que apresentaremos a seguir prioriza a consistência.

Nessa construção usamos uma função de incisão que protege entradas consistentes. Uma *função de incisão que protege entradas consistentes* "escolhe" pelo menos um elemento de cada conjunto minimal inconsistente procurando nunca escolher elementos da entrada A. Caso A seja inconsistente, haverá um conjunto minimal inconsistente tal que os elementos de A são seus únicos elemento. Nesse caso, A não é protegido.

Definição 8.7 (Função de incisão (σ) que protege entradas consistentes (A))**.** *Uma* função de incisão σ que protege entradas consistentes A *é definida como:*

1. $\sigma(A, B \perp\!\!\!\perp \Omega) \subseteq \bigcup(B \perp\!\!\!\perp \Omega)$.
2. *Se* $\emptyset \neq X \in B \perp\!\!\!\perp \Omega$, *então* $X \cap \sigma(A, B \perp\!\!\!\perp \Omega) \neq \emptyset$.
3. *Se A é consistente então* $A \cap \sigma(A, B \perp\!\!\!\perp \Omega) = \emptyset$.

A revisão é definida da mesma forma que na construção anterior:

$$B * A = B + A \setminus \sigma(A, B + A \perp\!\!\!\perp \Omega)$$

Porém, agora a função σ é uma função de incisão que protege entradas consistentes.

Essa construção é caracterizada pelos seguintes postulados de racionalidade para revisão bases de crenças: inclusão, pré-expansão, core-retainment, sucesso fraco e consistência forte. A consistência forte é garantida pela função de incisão que protege entradas consistentes, pois essa função sempre escolhe pelo menos um elemento de cada conjunto minimal inconsistente.

Teorema de Representação 8.8 (REKC)**.** *Uma* revisão externa kernel sem negação com consistência forte *(REKC)* $B * A$ *satisfaz sucesso fraco, consistência forte, inclusão, core-retainment e pré-expansão sse existe uma função de incisão* σ *que protege entradas consistentes tal que* $B * A = B + A \setminus \sigma(A, B + A \perp\!\!\!\perp \Omega)$.

Demonstração. Veja o apêndice B.4.2. □

8.1.2 Revisão Externa Partial Meet sem Negação

A *revisão externa partial meet sem negação* é definida de forma análoga a revisão kernel externa sem negação. Primeiro executamos uma expansão $B+A$ e em seguida removemos as inconsistências tomando cuidado de não remover a entrada.

A diferença entre as construções está na estratégia usada para remover as inconsistências. No caso da revisão kernel removemos pelo menos um elemento de cada conjunto minimal inconsistente. No caso da revisão partial meet escolhemos (através da função de seleção) alguns subconjuntos maximais consistentes de $B + A$ e pegamos sua intersecção.

Para proteger a entrada A e garantir alguma forma de sucesso, todos conjuntos maximais consistentes escolhidos pela função de seleção devem conter A. Se A é inconsistente, temos que escolher se privilegiaremos o postulado do sucesso ou da consistência. Isso acontece pois, nesse caso e apenas nesse caso, nenhum conjunto maximal consistente conterá A. Se optarmos por privilegiar, o sucesso o resultado da revisão será $B + A$. Se optarmos por privilegiar a consistência, então devemos escolher alguns elementos maximais consistentes independente de conter A.

A generalização da definição de *conjunto resíduo* foi apresentada na definição 6.7. Nesse capítulo usaremos essa mesma generalização em bases de crenças.

Revisão Externa Partial Meet sem Negação com Sucesso Forte (REPMS)

Para garantir o sucesso forte definiremos uma *função de seleção que proteja qualquer entrada.* Essa função possui dois parâmetros: o conjunto de sentenças a serem protegidas A e o conjunto resto $B \perp \Omega$. A função deve devolver alguns (pelo menos um) elementos de $B \perp \Omega$ que contenham A. Caso A seja inconsistente isso será impossível. Nesse caso a função deve devolver B. Formalmente temos:

Definição 8.9 (Função de seleção (γ) que protege a entrada (A))**.** *Uma* função de seleção γ que protege a entrada A *é definida como:*

1. $\emptyset \neq \gamma(A, B \perp \Omega) \subseteq B \perp \Omega$ *e* $A \subseteq \bigcap \gamma(A, B \perp \Omega)$ *se* A *é consistente.*
2. $\gamma(A, B \perp \Omega) = \{B\}$ *caso contrário.*

Exemplo 8.10: Considerando os conjuntos B e A do exemplo 8.5 temos que $B + A \perp \Omega$ é formado por nove elementos:

1. $B \setminus \{Saudável \sqsubseteq Alimento \sqcap \neg\exists rico_em.Gordura,$ $Saudável(margarina)\}$
2. $B \setminus \{Alimento \sqcap \exists ricoEm.Gordura(margarina),$ $\exists ricoEm.Trans(margarina), Saudável(manteiga)\}$
3. $B \setminus \{Alimento \sqcap \exists ricoEm.Gordura(margarina),$ $Trans \sqsubseteq Gordura, Saudável \sqsubseteq Alimento \sqcap \neg\exists rico_em.Trans,$ $Saudável(manteiga)\}$
4. $B \setminus \{Saudável(manteiga), Saudável(margarina)\}$
5. $B \setminus \{Alimento \sqcap \exists ricoEm.Gordura(manteiga),$ $Saudável(margarina)\}$

6. $B \setminus \{Alimento \sqcap \exists ricoEm.Gordura(margarina),$
   $Alimento \sqcap \exists ricoEm.Gordura(manteiga),$
   $\exists ricoEm.Trans(margarina)\}$
7. $B \setminus \{Saudável \sqsubseteq Alimento \sqcap \neg\exists rico_em.Gordura,$
   $Saudável \sqsubseteq Alimento \sqcap \neg\exists rico_em.Trans\}$
8. $B \setminus \{Saudável \sqsubseteq Alimento \sqcap \neg\exists rico_em.Gordura,$
   $\exists ricoEm.Trans(margarina)\}$
9. $B \setminus \{Alimento \sqcap \exists ricoEm.Gordura(margarina),$
   $Alimento \sqcap \exists ricoEm.Gordura(manteiga), Trans \sqsubseteq Gordura,$
   $Saudável \sqsubseteq Alimento \sqcap \neg\exists rico_em.Trans\}$

Os cinco primeiros não podem ser escolhidos por uma função de seleção γ que protege a entrada A, pois cada um desses conjuntos não contém pelo menos um elemento de A.

A *revisão externa partial meet sem negação com sucesso forte* é definida usando a função de seleção que protege a entrada γ como:

$$\bigcap \gamma(A, B + A \perp \Omega)$$

Esta revisão é completamente caracterizada pelos postulados: inclusão, pré-expansão, relevância, sucesso forte e consistência fraca. A relevância está associada com o método partial meet para remover inconsistências e o sucesso forte está associado com as restrições impostas à função de seleção.

Teorema de Representação 8.11 (REPMS)**.** *Em lógicas tarskianas e compactas, uma* revisão externa partial meet sem negação e com sucesso forte *(REPMS) $B * A$ satisfaz sucesso forte, consistência fraca, inclusão, relevância e pré-expansão sse existe uma função de seleção γ que protege a entrada tal que $B * A = \bigcap \gamma(A, B + A \perp \Omega)$.*

Demonstração. Veja o apêndice B.4.2. □

Revisão Externa Partial Meet sem Negação com Consistência Forte (REPMC)

A função de seleção que protege a entrada não garante que removeremos as inconsistências caso A seja inconsistente. Para garantir a consistência forte vamos definir uma *função de seleção que protege entradas consistentes*. Diferente da função de seleção apresentada na seção anterior, essa função sempre escolhe pelo menos um elemento do conjunto resto. Para garantir o sucesso fraco, impomos que, no caso em que A é consistente, a função seleciona só conjuntos maximais consistentes que contenham A.

Definição 8.12 (Função de seleção (γ) que protege entradas consistentes)**.** *Uma* função de seleção γ que protege entradas consistentes A *é definida como:*

1. $\emptyset \neq \gamma(A, B \perp \Omega) \subseteq B \perp \Omega$
2. *Se A é consistente então* $A \subseteq \bigcap \gamma(A, B \perp \Omega)$

A *revisão externa partial meet sem negação com consistência forte* é definida também como:

$$\bigcap \gamma(A, B + A \perp \Omega)$$

Porém, agora γ é uma função de seleção que protege entradas consistentes.

Os postulados que caracterizam essa revisão, com exceção do sucesso e da consistência são os mesmo que caracterizam a revisão da seção anterior. A revisão apresentada aqui satisfaz o sucesso fraco e a consistência forte por causa das restrições impostas à função de seleção.

Teorema de Representação 8.13 (REPMC)**.** *Em uma lógica tarskiana e compacta, uma* revisão externa partial meet sem negação e com consistência forte *(REPMC)* $B * A$ *satisfaz sucesso fraco, consistência forte, inclusão, relevância e pré-expansão sse existe uma função de seleção* γ *que protege entradas consistentes tal que* $B * A = \bigcap \gamma(A, B + A \perp \Omega)$*.*

Demonstração. Veja o apêndice B.4.2. □

8.2 Revisão Interna sem Negação

Para construir a revisão externa adicionamos A à base B e em seguida removemos as inconsistências. Na revisão interna primeiro "abrimos espaço" para α e em seguida o adicionamos. Formalmente a construção da revisão interna é dada pela identidade de Levi [Han93, AGM85]:

$$B * A = B - \neg A + A$$

Como na revisão externa, na revisão interna dependemos da existência da negação de A. Nessa seção vamos apresentar construções para a revisão interna que não usam a negação. A estratégia usada aqui será a mesma da Seção 7: primeiro removemos sentenças que são inconsistentes junto de A e em seguida acrescentamos A.

Para garantir a minimalidade da revisão devemos remover a menor quantidade de informação possível ao "abrir espaço" para A. Para isso podemos usar subconjuntos maximais de B que são consistentes juntos de A, como na Seção 7 (revisão interna partial meet sem negação), ou podemos definir conjuntos minimais que junto de A são consistentes (revisão interna kernel sem negação).

Com relação aos postulados que caracterizam as construções, a revisão interna difere da externa por não satisfazer a pré-expansão. Além disso, a uniformidade é satisfeita pela revisão interna.

A generalização do postulado da uniformidade é natural:

(uniformidade) Se para todo $B' \subseteq B$ temos que $B' \cup A$ é inconsistente sse $B' \cup A'$ é inconsistente, então $B \cap B * A = B \cap B * A'$.

Nas próximas seções apresentaremos as duas formas de construir a revisão interna sem negação e provaremos os respectivos teoremas de representação.

8.2.1 Revisão Interna Partial Meet sem Negação (RIPM)

Definiremos a construção da *revisão interna partial meet sem negação* de forma muito similar à construção da revisão AGM sem negação (Seção 7). Primeiro removemos algumas sentenças usando a construção $B \downarrow A$ e, então, adicionamos A. Definimos a revisão como:

$$B * A = (B \downarrow A) \cup A$$

A única diferença dessa construção para a construção apresentada na Seção 3.3 é que aqui o conjunto B é uma base, ou seja, não é necessariamente fechado por consequência lógica.

> **Exemplo 8.14:** Considerando os conjuntos B e A como no exemplo 8.5 temos que $B \downarrow A$ formado exatamente pelos últimos quatro elementos do exemplo 8.10.

Essa construção é caracterizada pelos seguintes postulados: sucesso forte, consequência fraca, inclusão, relevância e uniformidade. Com exceção do fecho, esses são os mesmos postulados que caracterizam a revisão AGM sem negação.

Para revisão AGM sem negação, provamos que a equivalência entre postulados e construção vale sempre que a lógica subjacente for distributiva, tarskiana e satisfaça a explosão fraca. Em bases de crenças podemos mostrar que o teorema de representação vale para qualquer lógica monotônica e compacta. Ou seja, em bases de crenças o resultado vale para uma classe maior de lógicas.

Teorema de Representação 8.15 (RIPM)**.** *Seja a lógica subjacente tarskiana e compacta, a revisão $*$ satisfaz inclusão, sucesso forte, consistência fraca, relevância e uniformidade sse existe alguma função de seleção γ tal que $B * A = \bigcap \gamma(B \downarrow A) \cup A$*

Demonstração. Veja o apêndice B.4.3. □

8.2.2 Revisão Interna Kernel sem Negação (RIK)

Como na construção da revisão interna partial meet sem negação, a idéia da revisão interna kernel sem negação consiste em primeiro remover sentenças da base e em seguida acrescentar A. As sentenças que serão removidas são aquelas que pertencem a algum subconjunto minimal de B que junto de A seja inconsistente (escrevemos $B{\downdownarrows}A$).

Definição 8.16. *Os* subconjuntos minimais de B que junto de A são inconsistentes *formam o conjunto $B{\downdownarrows}A$ que é definido como $X \in B{\downdownarrows}A$ sse:*

1. *$X \subseteq B$.*
2. *$X \cup A$ é inconsistente.*
3. *Se $X' \subset X$ então $X' \cup A$ é consistente.*

A função de incisão é definida de forma usual:

Definição 8.17. *Uma função σ é uma função de incisão sse:*

1. *$\sigma(B{\downdownarrows}A) \subseteq \bigcup B{\downdownarrows}A$*
2. *Se $\emptyset \neq X \in B{\downdownarrows}A$ então $X \cap \sigma(B{\downdownarrows}A) \neq \emptyset$*

Exemplo 8.18: Considerando os conjuntos B e A do exemplo 8.5 temos que $B{\downdownarrows}A$ é formado pelos conjuntos:

$\{\exists rico_em.Trans(margarina),$
$\quad Saudável \sqsubseteq Alimento \sqcap \neg\exists rico_em.Trans\}$
$\{\exists rico_em.Trans(margarina),$
$\quad Trans \sqsubseteq Gordura,$
$\quad Saudável \sqsubseteq Alimento \sqcap \neg\exists rico_em.Gordura\}$
$\{\exists rico_em.Gordura(margarina),$
$\quad Saudável \sqsubseteq Alimento \sqcap \neg\exists rico_em.Gordura\}$
$\{\exists rico_em.Gordura(manteiga),$
$\quad Saudável \sqsubseteq Alimento \sqcap \neg\exists rico_em.Gordura\}$

A *revisão interna kernel sem negação* (RIK) é definida da seguinte forma:

$$B * A = (B \setminus \sigma(B{\downdownarrows}A)) \cup A$$

Ou seja, removemos pelo menos uma sentença de cada elemento de $B{\downdownarrows}A$ e em seguida adicionamos A.

A diferença entre a RIPM e a RIK é análoga a diferença entre as revisões kernel e partial meet da Seção 8.1. Não surpreendentemente, os postulados que caracterizam RIK são os mesmos que caracterizam RIPM trocando a relevância pelo core-retainment. Ou seja, essa revisão é completamente caracterizada pelos seguintes postulados: sucesso forte, consistência fraca, inclusão, uniformidade e core-retainment. Como em RIPM, o teorema de representação vale para qualquer lógica tarskiana e compacta:

Teorema de Representação 8.19 (RIK)**.** *Seja a lógica subjacente tarskiana e compacta, a operação de revisão $*$ satisfaz sucesso forte, consistência fraca, inclusão, uniformidade e core-retainment sse $B * A = (B \setminus \sigma(B{\downdownarrows}A)) \cup A$ para alguma função de incisão σ.*

Demonstração. Veja o apêndice B.4.3. □

8.3 Conclusão

Postulados	REKS	REKC	REPMS	REPMC	RIK	RIPM
sucesso	forte	fraco	forte	fraco	forte	forte
consistência	fraca	forte	fraca	forte	fraca	fraca
inclusão	sim	sim	sim	sim	sim	sim
pré-expansão	sim	sim	sim	sim	não	não
minimalidade	core-ret.	core-ret.	relev.	relev.	core-ret.	relev.
uniformidade	não	não	não	não	sim	sim

Tabela 8.1: Revisão em bases de crenças sem negação

As construções para revisão em bases de crenças apresentadas na Seção 3.3 dependem da negação da entrada A. Diversas lógicas, porém, não possuem uma negação natural para ser usada nessas construções. Uma solução para esse problema seria usar a semi-revisão [Han97]. A semi-revisão, porém, não satisfaz o sucesso. Nesse capítulo mostramos como revisar bases de crenças sem usar a negação. Os autores de [HWKP06] estudaram a aplicação da semi-revisão em lógicas de descrição para bases de crenças.

Nesse capítulo mostramos construções para revisão em bases de crenças que não dependem da negação de sentenças. Mostramos o teorema de representação para cada uma delas e provamos que esses teoremas valem para qualquer lógica tarskiana e compacta.

Outro trabalho relacionado a este é [QHHP08]. Nesse trabalho os autores estudam a revisão de bases de crenças em terminologias usando uma abordagem que utiliza operações cujo critério de minimalidade depende da cardinalidade dos subconjuntos da base que implicam a entrada.

Tanto a contração quanto a revisão em bases de crenças se relacionam intimamente ao problema da depuração de ontologias [SC03, Kal06]. Os resultados apresentados nessa seção formam uma base teórica para o estudo de técnicas de depuração de ontologias. Em particular, os teoremas de representação mostram quais postulados caracterizam precisamente cada construção. Dessa forma as construções podem ser comparadas de um ponto de vista mais abstrato. No capítulo 9 mostraremos o outro lado dessa moeda, ou seja, como usar algoritmos para depuração de ontologias para implementar as construções apresentadas nessas seção.

Capítulo 9

Implementação das Operações

Nos capítulos anteriores mostramos como adaptar as operações em conjuntos de crenças e em bases de crenças de forma que elas sejam aplicáveis a uma gama maior de lógicas. Estávamos particularmente interessados nas lógicas de descrição por sua aplicação em web semântica. Nesse capítulo mostraremos como implementar parte das construções propostas nos capítulos anteriores.

Mostraremos algoritmos para construções em bases de crenças, ou seja, para as construções apresentadas no capítulo 8. Não trataremos de algoritmos para conjuntos de crenças.

As construções que propusemos para revisão em bases de crenças baseiam-se em conjuntos kernel ou conjuntos resíduo. Mostraremos como computar ambos os conjuntos.

Os algoritmos apresentados aqui são conhecidas na área de depuração de ontologias [PSK05, KPSH05, SC03]. Nossas contribuições nessa área foram:

- mostrar a relação entre esses algoritmos e a teoria de bases de crenças
- implementar esses algoritmos em um plug-in para o editor de ontologias Protégé[1].

Mostraremos primeiro dois algoritmos de [Kal06] para computar um elemento do kernel (seção 9.1). O primeiro, chamado de black-box, faz chamadas ao motor de inferências sem mexer em sua estrutura interna, ou seja, trata-o como uma caixa preta. O segundo altera a estrutura interna do motor de inferências e, por isso, é chamado de glass-box. Em seguida apresentaremos a idéia de [Was00a] que usa o algoritmo de Reiter [Rei87] para encontrar os demais elementos do kernel (seção 9.2). Na seção 9.3 mostraremos algoritmos para computar um conjunto resíduo a partir de um conjunto kernel e vice-versa sem fazer chamadas ao motor de inferência. Por fim, na seção 9.4, mostraremos alguns detalhes da implementação de um plug-in para o Protégé que computa as operações apresentadas no capítulo 8.

9.1 Algoritmos Glass-Box e Black-Box

O algoritmo black-box para encontrar um elemento de $B \perp\!\!\!\perp \alpha$ é baseado na estratégia expande-encolhe [Kal06]. Esse algoritmo consiste em duas partes: na primeira (expande) sentenças de B são adicionadas em B' uma por vez até que $B' \vdash \alpha$; na segunda (encolhe) é retirado um elemento de B' por vez e em cada passo é testado se esse conjunto ainda prova α. Esse algoritmo é chamado também de *black-box* porque ele não depende do provador de teoremas usado.

O algoritmo 9.1 chama o provador de teoremas $2(\#B)$ vezes no pior caso. Uma heurística para diminuir esse número de chamadas deve tentar encontrar o menor conjunto $B' \subseteq B$ possível tal que $B' \vdash \alpha$.

Kalyanpur apresenta em [Kal06] o *algoritmo glass-box*, que possui esse nome pois depende do provador de teoremas usado. A idéia do algoritmo consiste em marcar as sentenças usadas para provar α a partir de B.

[1] http://protege.stanford.edu

Algoritimo 9.1 Algoritmo para achar um elemento do kernel.

```
BLACK-BOX(B, α)
 1  ▷ Expande
 2  B' ← ∅
 3  for β ∈ B
 4        do B' ← B' ∪ {β}
 5           if α ∈ Cn(B')
 6              then devolva B'
 7  ▷ Encolhe
 8  for β ∈ B'
 9        do if α ∈ Cn(B' \ {β})
10              then B' ← B' \ {β}
11  devolva B'
```

O algoritmo começa com um ABox $\mathcal{A}$ em que cada fórmula ($D(b)$) é marcada com $\emptyset$ ($D(b)^{\emptyset}$). A marcação deve agregar os axiomas usados na derivação do conflito. Para tanto, é preciso adaptar o método de tableaux apresentado na seção 4.1.3. Na figura 9.2 mostramos o tableau adaptado para $\mathcal{ALC}$ como apresentado em [SC03]. Kalyanpur mostra em [Kal06] como estender esse algoritmo para a lógica $\mathcal{SHOIN}(\mathbf{D})$.

Algoritimo 9.2 Algoritmo glass-box para $\mathcal{ALC}$

regra $\rightarrow GCI$**:**

condição: $\mathcal{A}$ contém $C(x)^{label}$ e $C \sqsubseteq D$

ação: $\mathcal{A} := \mathcal{A} \cup \{C(x)^{label \cup \{C \sqsubseteq D\}}\}$.

regra $\rightarrow \sqcap$**:**

condição: $\mathcal{A}$ contém um indivíduo $C_1 \sqcap C_2(x)^{label}$, mas não contém ambos $C_1(x)^{label}$ e $C_2(x)^{label}$.

ação: $\mathcal{A} := \mathcal{A} \cup \{C_1(x)^{label}, C_2(x)^{label}\}$.

regra $\rightarrow \sqcup$**:**

condição: $\mathcal{A}$ contém $C_1 \sqcup C_2(x)^{label}$, mas não contém $C_1(x)^{label}$ nem $C_2(x)^{label}$.

ação: cria o ABox $\mathcal{A}' := \mathcal{A} \cup \{C_1(x)^{label}\}$ e $\mathcal{A} := \mathcal{A} \cup \{C_2(x)^{label}\}$

regra $\rightarrow \exists$**:**

condição: $\mathcal{A}$ contém $\exists R.C(x)^{label}$ e $\forall R.C_1(x)^{label_1} \ldots \forall R.C_n(x)^{label_n}$ e todas outras regras já foram aplicadas para x.

ação: $\mathcal{A} := \mathcal{A} \cup \{C(y)^{label}, C_1(y)^{label \cup label_1}, \ldots, C_n(y)^{label \cup label_n}\}$ em que y é um novo indivíduo.

Sabemos que esse conjunto B' de sentenças marcadas é suficiente para provar α. Logo, se usarmos B' como entrada para o algoritmo expande-encolhe esse algoritmo fará menos chamadas ao provador de teoremas.

9.2 Algoritmo de Reiter

Para encontrar os demais elementos do kernel, Kalyanpur [Kal06] usa uma idéia de [Was00a] que, por sua vez, usa uma adaptação do algoritmo de Reiter mostrado no algoritmo 9.2. O algoritmo de Reiter encontra os cortes mínimos de uma classe[2] de conjuntos.

Definição 9.1 (Corte Mínimo)**.** *Um* corte *em uma classe $\mathcal{B}$ de conjuntos é um conjunto B cuja intersecção com cada conjunto da classe $\mathcal{B}$ não é vazia. Um corte C é* mínimo *sse não existe nenhum corte C' propriamente contido em C.*

O algoritmo de Reiter acha os cortes mínimos de uma classe de conjuntos. O algoritmo 9.2 apresentado abaixo é uma simplificação do algoritmo de Reiter [Rei87].

Algoritimo 9.3 Algoritmo de Reiter para computar cortes mínimos em uma classe de conjuntos.

```
REITER(B)
 1  Corte ← ∅
 2  fila ← uma fila vazia
 3  S ← um elemento qualquer de B
 4  for s ∈ S
 5        do insira {s} no começo da fila.
 6  while fila não está vazia
 7        do Hn ← último elemento da fila
 8           remove o último elemento da fila
 9           if ∃C ∈ Corte tal que C ⊆ Hn
10              then continue
11           elseif ∃S ∈ B tal que Hn ∩ S = ∅
12              then for s ∈ S
13                         do insira Hn ∪ {s} na fila
14           else
15                    Corte ← Corte ∪ {Hn}
16  devolva Corte
```

A variável Hn recebe um potencial corte minimal da fila. Caso exista algum corte que contém Hn então Hn não é minimal (linhas 9 e 10) e podemos ignorá-lo. Caso contrário verificamos se existe algum elemento de $\mathcal{B}$ cuja intersecção com Hn ainda é vazia. Se não existir nenhum então Hn é um corte minimal e então o acrescentamos ao conjunto $Corte$ (linha 15) . Caso contrário, para cada elemento s do conjunto que não intersecciona Hn temos que $Hn \cup \{s\}$ é um potencial corte mínimo de $\mathcal{B}$. Portanto, inserimos cada um desses $Hn \cup \{s\}$ na fila para verifica-los numa futura iteração do laço. Ao fim desse processo o conjunto $Corte$ contém todos os cortes mínimos de $\mathcal{B}$.

Para achar os demais elementos do kernel aplicamos uma adaptação do algoritmo de Reiter. A idéia de adaptar o algoritmo de Reiter para achar os demais elementos do kernel foi proposta primeiramente em [Was00b] e aplicada em lógicas de descrição no contexto de depuração de ontologias em [Kal06]. O algoritmo 9.2 é uma versão simplificada desses algoritmos.

Esse algoritmo começa calculando um elemento do kernel usando algum dos algoritmos apresentados na seção anterior. A cada iteração calculamos o menor subconjunto S de $B \setminus Hn$ que implica α. Se não existe subconjunto de $B \setminus Hn$ então Hn é um corte no conjunto kernel (linha 19). Caso contrário, S é um elemento de $K \perp\!\!\!\perp \alpha$.

[2] Usamos o termo classe simplesmente como um conjunto de conjuntos.

Algoritimo 9.4 Algoritmo para computar os demais elementos do kernel.

```
KERNEL(B, α)
 1  Corte ← ∅
 2  fila ← uma fila vazia
 3  S ← BLACK-BOX(B, α)
 4  Kernel ← {S}
 5  for s ∈ S
 6        do insira {s} no começo da fila.
 7  while fila não está vazia
 8        do Hn ← último elemento da fila
 9           remove o último elemento da fila
10           if ∃C ∈ Corte tal que C ⊆ Hn
11              then continue
12           elseif α ∈ Cn(B \ Hn)
13              then S ← BLACK-BOX(B \ Hn, α)
14                   Kernel ← Kernel ∪ {S}
15                   for s ∈ S
16                        do insira Hn ∪ {s} na fila
17           else
18                   Corte ← Corte ∪ {Hn}
19  devolva Kernel
```

9.3 Relações entre o Conjunto Kernel e o Conjunto Resíduo

Uma forma de se obter o conjunto resíduo é extraí-lo do kernel. Isso é possível porque o conjunto resíduo $B \perp \alpha$ se identifica com o conjunto dos complementos entre B e os elementos do kernel $B \perp\!\!\!\perp \alpha$. Da mesma forma, podemos calcular o kernel a partir do conjunto resíduo[3]. O teorema a seguir formaliza esses resultados:

Teorema 9.2. *Seja $\mathscr{L}$ uma lógica tarskiana e compacta, $B \subseteq \mathscr{L}$ e $\alpha \in \mathscr{L}$:*

1. *$X \in \{B \setminus Y : Y \in B \perp \alpha\}$ sse X é um corte mínimo de $B \perp\!\!\!\perp \alpha$*
2. *$X \in B \perp\!\!\!\perp \alpha$ sse X é um corte mínimo de $\{B \setminus X : X \in B \perp \alpha\}$*

Demonstração. Veja o apêndice B.5 □

Desta forma, basta calcular um dos dois conjuntos (kernel ou resíduo). O outro pode ser inferido sem novas chamadas ao provador de teoremas, usando o algoritmo de Reiter. Esse resultado sugere que ambos os problemas possuem características parecidas. Porém, os seguintes exemplos mostram que dependendo do problema pode ser bem mais fácil calcular o kernel ou o conjunto resíduo. Isso ocorre pois, o tamanho do conjunto kernel e do conjunto resíduo podem variar muito.

Exemplo 9.3:

$$\begin{aligned} B \;=\; & \{b_1, b_1 \to a, \\ & b_2, b_2 \to a, \\ & \ldots, \\ & b_n, b_n \to a\} \end{aligned}$$

[3] A relação entre o conjunto kernel e o conjunto resíduo, mais precisamente a relação entre a função de incisão e a função de seleção foi estudada pelos autores de [FFKI06].

Nesse exemplo $B \perp a$ possui um número de elementos proporcional a 2^n enquanto $B \perp\!\!\!\perp a$ possui um número de elementos proporcional a n.

Exemplo 9.4:

$$\begin{aligned} B \;=\; & \{b_1, b_1', \\ & (b_1 \vee b_1') \to b_2, (b_1 \vee b_1') \to b_2', \\ & (b_2 \vee b_2') \to b_3, (b_2 \vee \underline{2}') \to b_3', \\ & \ldots, \\ & bn \vee b_n' \to a\} \end{aligned}$$

Nesse caso o conjunto $B \perp a$ possui um número de elementos proporcional a n enquanto $B \perp\!\!\!\perp a$ dessa vez possui um número de elementos proporcional a 2^n.

9.4 Implementação

Implementamos as construções da seção 8.1 usando os algoritmos das seções 9.1 e 9.2 como um plug-in para o *Protégé* 4.0. O Protégé é um programa de código aberto usado para ajudar os engenheiros de ontologia a modelar ontologias que podem ser exportadas para as linguagens OWL. O programa oferece uma interface gráfica intuitiva e compatibilidade com diversos provadores de teoremas conhecidos como Pellet [SPG$^+$07], o FaCT++ [TH06] e o Hermit [MSH07].

Além do Protégé 4.0, usamos a API OWL[4] [BVL03] para Java. Essa API permite definirmos de forma simples conceitos e axiomas em OWL e vem integrada com o Pellet[5] que usamos como motor de inferência. Além disso, a API OWL implementa o algoritmo glass-box mostrado nesse capítulo.

O plug-in que desenvolvemos permite ao usuário aplicar uma contração ou uma revisão à ontologia sendo desenvolvida. Em ambos os casos o usuário pode escolher quais dos postulados do capítulo 8 ele gostaria de satisfazer. Uma vez escolhidos os postulados o usuário pode introduzir uma sentença que ele gostaria de contrair ou revisar (veja a figura 9.2). O programa calculará o kernel ou o conjunto resíduo, dependendo se o usuário escolheu satisfazer core-retainment ou relevância. O usuário pode, então escolher quais sentenças prefere remover (veja as figuras 9.4, 9.5). Por fim, o programa efetua as remoções e o usuário pode recomeçar o processo (veja as figuras 9.6, 9.7).

A forma como a OWL API foi projetada (centrada na lógica, nos axiomas etc.) facilitou bastante a implementação que transcorreu sem grandes dificuldades. A maior dificuldade foi desenvolver em uma versão alfa do Protégé 4 que se alterou muito durante o projeto. Por este motivo, nossa implementação terminou incompatível com sua versão estável.

O programa foi testado com sucesso usando alguns exemplos muito simples como o da manteiga e da margarina apresentado nesse trabalho (veja as figuras abaixo). Os testes não foram suficientes para garantir a usabilidade do programa principalmente quanto a sua escalabilidade.

[4] http://owlapi.sourceforge.net/
[5] http://clarkparsia.com/pellet/

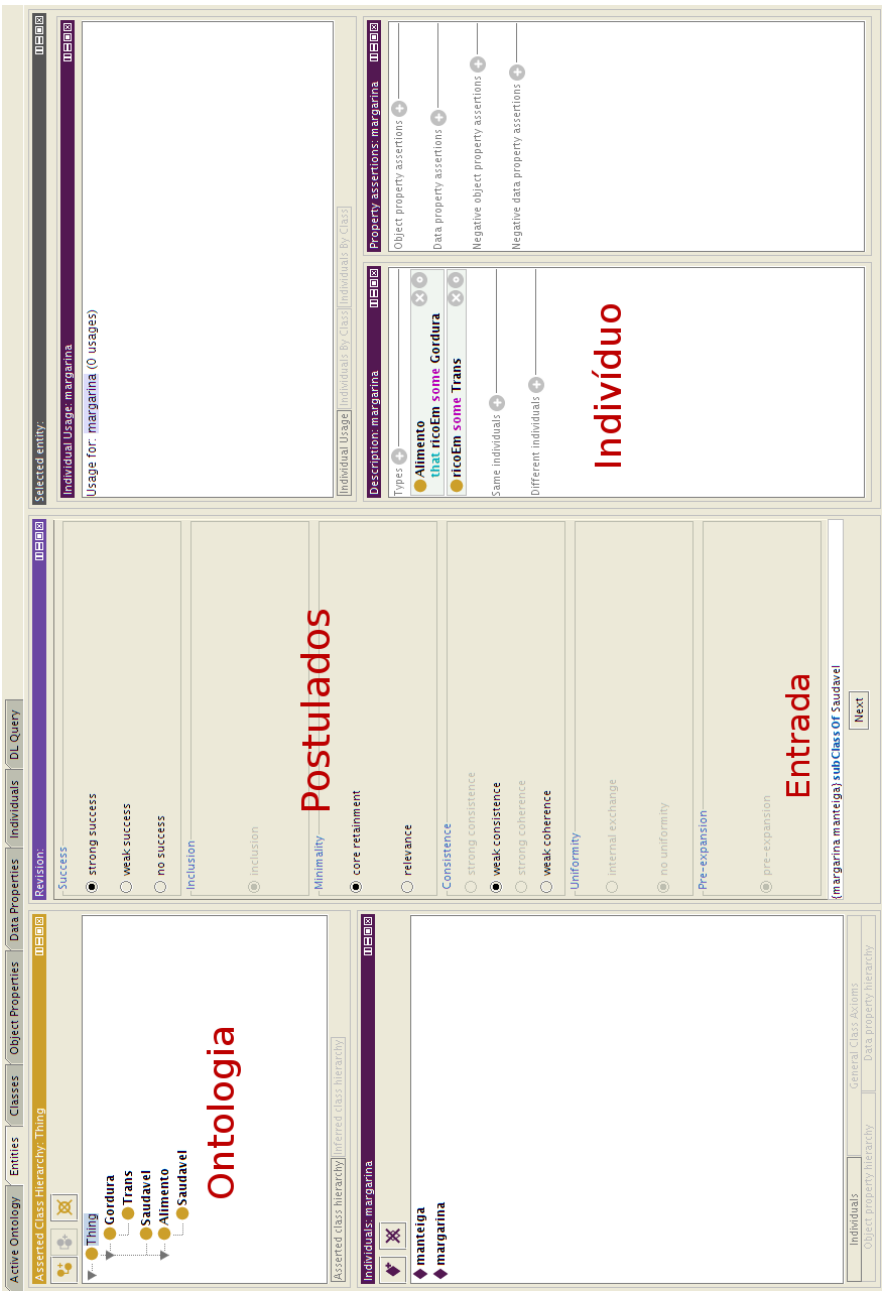


Figura 9.1: Visão geral do Protégé com o plug-in para revisão.

9.5 Conclusão

Para resolver o problema da revisão e da contração em bases de crenças é preciso calcular o kernel ou o conjunto resíduo de uma base. Mostramos que é possível calcular o kernel de uma bases de crenças em lógicas de descrição usando algoritmos conhecidos da área de depuração de ontologias. Uma vez calculado o kernel é possível calcular o conjunto resíduo usando o algoritmo de Reiter sem precisar fazer novas chamadas ao provador de teoremas.

Nossa principal contribuição nessa área foi mostrar a relação entre estes algoritmos e os problemas de revisão em bases de crenças. Assim fortalecemos ambas as áreas, pois de um lado a área de depuração pode ser avaliada de um ponto de vista abstrato dos postulados de racionalidade para as operações e de outro a área de revisão pode se utilizar dos algoritmos já conhecidos para depuração de ontologias.

O problema de calcular o kernel de uma base, assim como de calcular o conjunto resíduo, é intrinsicamente exponencial, pois o próprio resultado pode ter tamanho exponencial. A complexidade de tais problemas é usualmente medida pela complexidade do intervalo entre as soluções. Em [PS10a, PS10b] os autores mostram a complexidade do kernel para diversas lógicas de descrição polinomiais.

Implementamos esses algoritmos como um plugin para o Protégé 4 e os testamos minimamente. A implementação foi bastante prejudicada pelo fato de estarmos lidando como uma versão instável do Protégé, mas cumpriu o objetivo de mostrar que é possível implementar as construções para bases de crenças apresentadas no capítulo 8. Os testes, porém, não foram suficiente para concluirmos nada sobre a escalabilidade da implementação. Além disso, hoje a implementação é incompatível com as versões estáveis do Protégé 4.

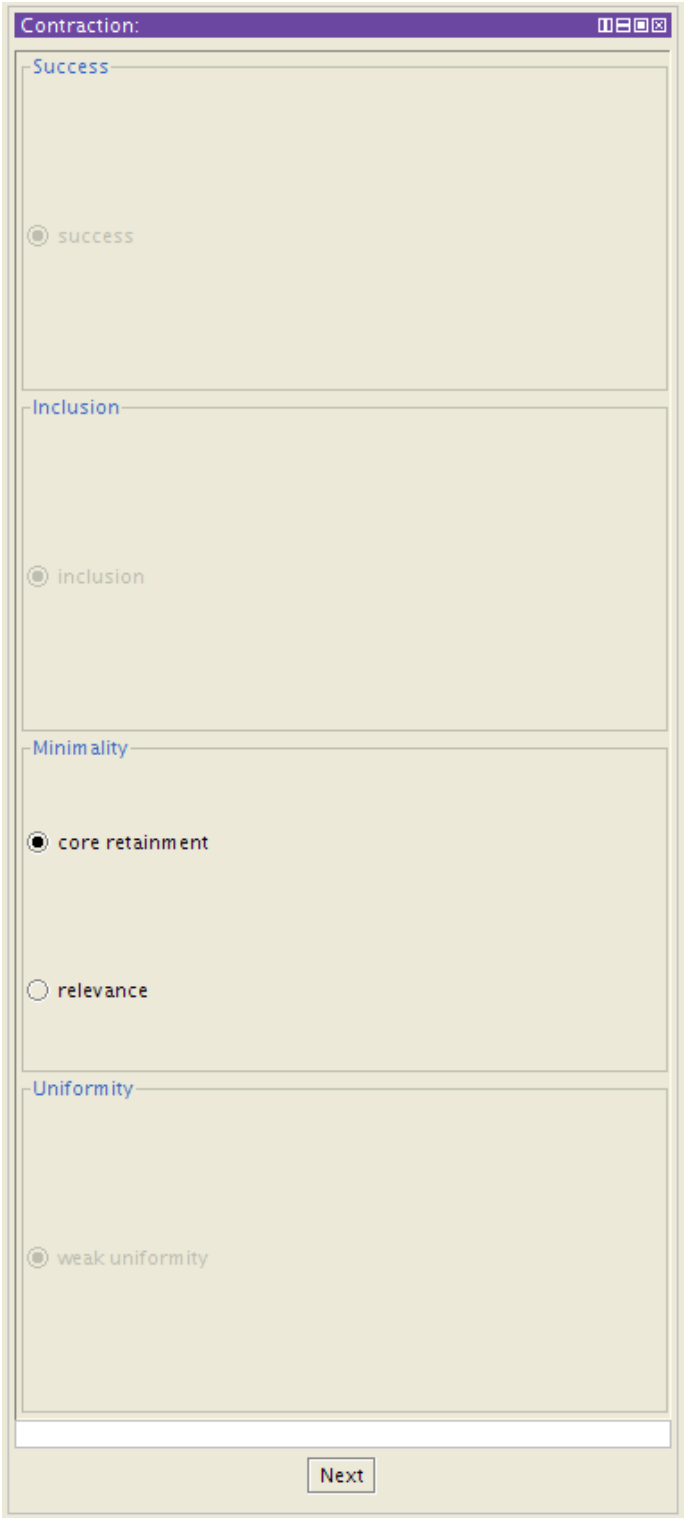


Figura 9.2: Opções do plug-in para contração.

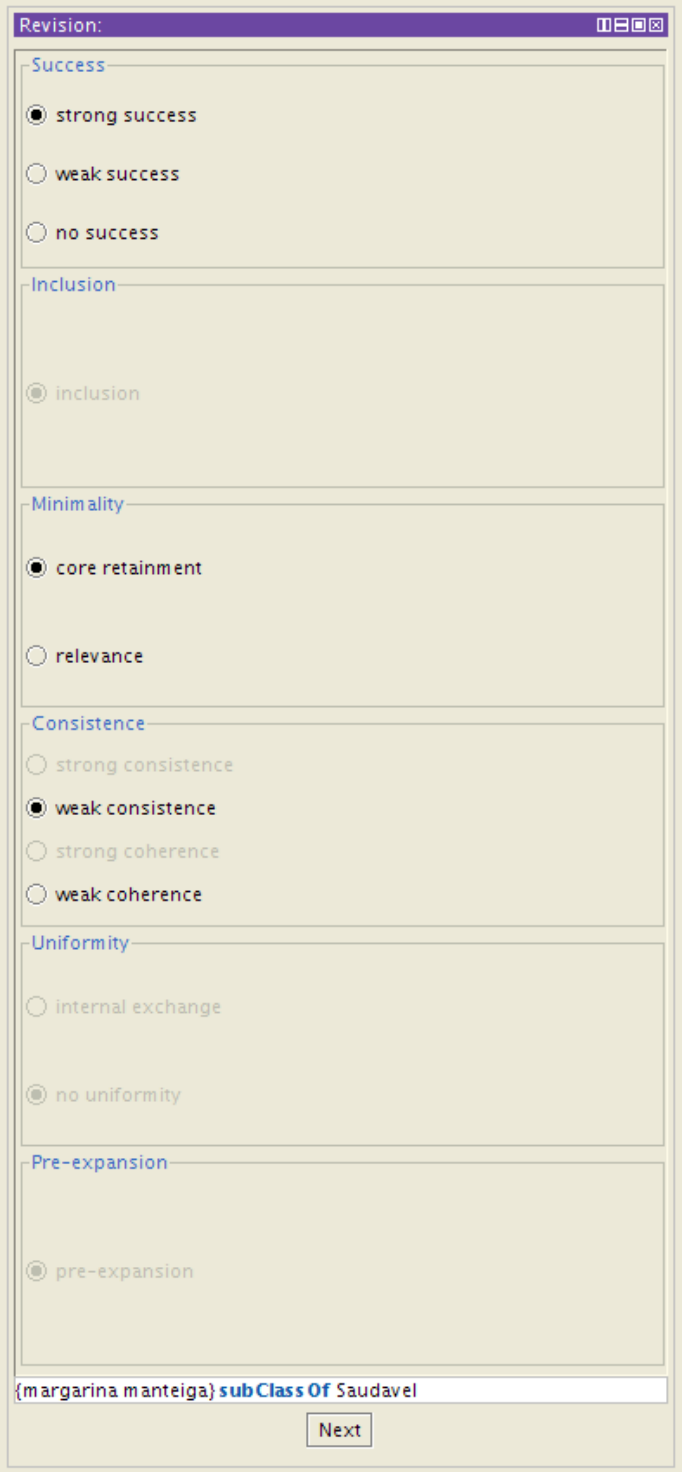


Figura 9.3: Opções do plug-in para revisão.

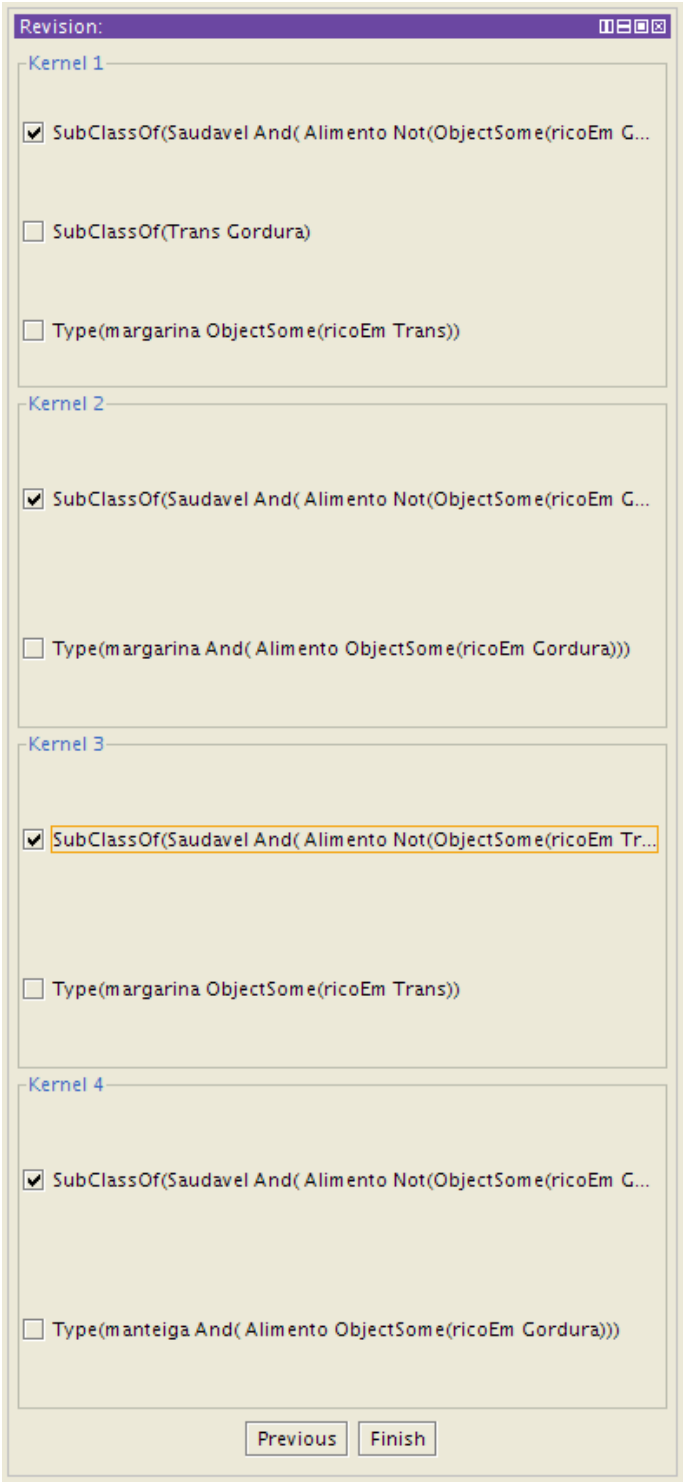


Figura 9.4: Elementos do conjunto kernel.

Figura 9.5: Elementos do conjunto resíduo.

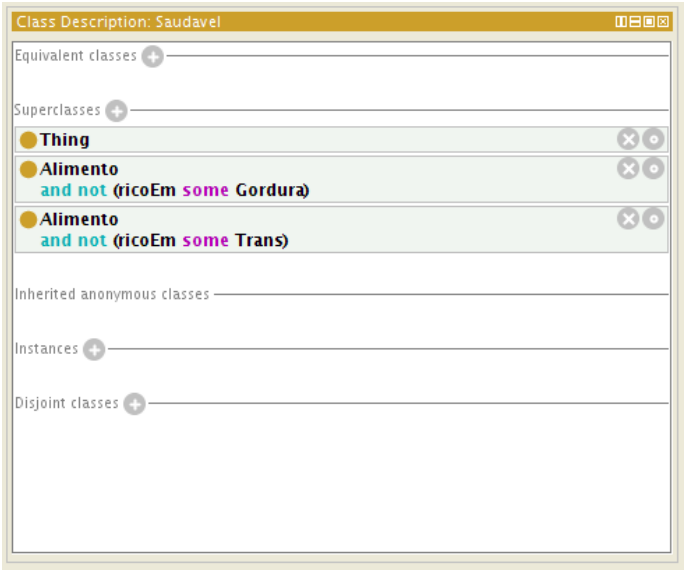


Figura 9.6: O conceito *Saudável* antes da revisão.

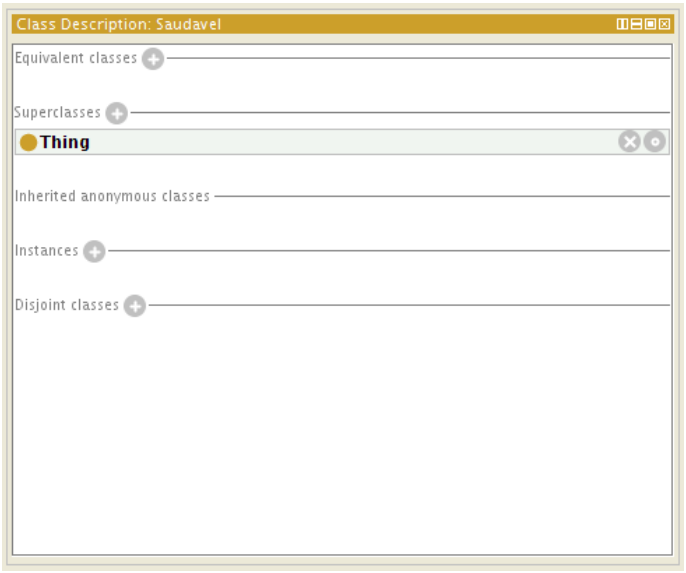


Figura 9.7: O conceito *Saudável* depois da revisão.

Capítulo 10

Conclusões e Trabalhos Futuros

10.1 Conclusões

Diversos autores apontam para a importância do estudo da dinâmica das ontologias [HS04, Sto04, Flo06]. Como a teoria da revisão de crenças há tempos tem estudado a dinâmica das bases de conhecimento, aplicá-la às lógicas usadas para representar ontologias (lógicas de descrição) parece uma idéia promissora. Apesar das dificuldades apontadas por Flouris em [Flo06], mostramos que é possível aplicar a teoria de revisão de crenças às lógicas de descrição. Para tanto precisamos adaptar os postulados e as construções para revisão de crenças. Essa generalização pode ser útil inclusive para aplicar a revisão de crenças em outras lógicas como a lógica de Horn e a lógica intuicionista.

A interação entre a área de revisão de crenças as lógicas de descrição provou-se fértil. Descobrimos que determinadas propriedades lógicas como a distributividade, o fecho por negação, a decomponibilidade e a compacidade são essenciais para compreender a teoria da revisão de crenças. Além disso, estudamos como essas propriedades se relacionam (figura 5.7).

Investigamos o que ocorre ao trocarmos o postulado da recuperação pela relevância nos postulados AGM para contração. Esse novo conjunto de postulados é compatível com uma classe maior de lógicas. Além disso mostramos que ele é mais adequado para caracterizar a contração partial meet. Estes resultados foram apresentados em [RW06], [RW09a] e [RWA$^+$09]. Mostramos também que em lógicas booleanas a relevância e a recuperação são equivalentes, mas para tantas outras isso não vale. Nessa segunda classe de lógicas a relevância e a recuperação postulados distintos. Portanto, nesses casos o conceito de minimalidade é completamente distinto do conceito de recuperabilidade. A tabela 6.1.3 resume os resultados sobre esse conjunto de postulados: os postulados para contração-relevante.

Definimos uma revisão que não utiliza a identidade de Levi e não depende da negação. Essa construção satisfaz os postulados AGM para revisão. Apresentamos, além disso, um conjunto de postulados que caracteriza essa construção em qualquer lógica distributiva e compacta. Publicamos esse teorema de representação em [RW09a]. Exemplos de lógicas distributivas e compactas foram apresentados no capítulo 5 e publicadas em [RW10].

Em relação as lógicas de descrição, aprendemos que elas possuem características bastante peculiares. Muitas delas não são fechadas por negação, não são compatíveis com os postulados AGM e não são distributivas. É interessante notar que algumas dessas peculiaridades, a incompatibilidade AGM por exemplo, não ocorrem justamente nas lógicas de descrição que são equivalentes a lógicas modais.

Estendemos o trabalho de Flouris provando que outras lógicas como a lógica de Horn e a lógica intuicionista também são incompatíveis com os postulados AGM para contração. Além disso, mostramos quais propriedades cada uma dessas lógicas satisfaz (tabela 5.1).

Em [RW10] mostramos quais lógicas de descrição são distributivas e quais são fechadas por negação. A relação entre as propriedades lógicas e a análise da lógica de Horn e da lógica intuicionista ainda não foram publicados.

Estudamos também a revisão em bases de crenças. Nessa área definimos seis tipos de revisão junto do conjunto de postulados que as caracterizam. Nenhuma dessas construções depende da definição da negação e pode ser aplicadas a qualquer lógica tarskiana e compacta. Esses resultados estão resumidos na tabela 8.1 e foram publicados em [RW07] e [RW09b].

Os algoritmos usados na área de depuração de ontologias podem ser usados para as construções em bases de crenças. Assim, por um lado temos algoritmos para as operações em bases de crenças. Por outro, a teoria de revisão em bases de crenças pode ser usada como base teórica para estudar esses algoritmos. Além disso, implementamos essas operações como um plug-in para o Protégé. Apresentamos essa implementação em [RW08b]. Esse trabalho deverá prosseguir com participação de outro alunos.

Em [RW08c] mostramos como inferir um determinado conjunto resíduo $B \perp \alpha$ partindo de um kernel $B \perp\!\!\!\perp \alpha$ sem fazer chamadas ao provador de teoremas. Nesse trabalho argumentamos que a contração partial meet deveria ser mais eficiente que a contração kernel. Uma comparação mais rigorosa está sendo feita por alunos do grupo.

No apêndice A mostraremos que ao relaxar o postulado da inclusão na contração em bases de crenças obtemos diferentes graus de recuperação que podem ser úteis para tratarmos, por exemplo, de exceções. Esse resultado foi publicado em [RW08c].

Durante a concepção da tese desenvolvemos colaborativamente um outro plug-in para o Protégé que classifica uma ontologia quanto a lógica de descrição mais adequada para representá-la [KPTR$^+$06].

10.2 Trabalhos Futuros

Certamente o trabalho desenvolvido nesta tese não esgota o tema abordado. Em seguida listamos algumas possibilidades de trabalhos futuros que se relacionam com os temas abordados aqui.

1. Generalização da construção para revisão de crenças AGM em lógicas não distributivas.
2. Estudo da contração AGM em lógicas não compactas.
3. Implementação e testes mais rigorosos do plug-in desenvolvido.
4. Adaptação do plug-in para outras plataformas como Jena[1] e para outras lógicas como a LPC.
5. Comparação mais rigorosa da eficiência em computar o conjunto kernel e o conjunto resíduo.
6. Estudo das lógicas modais híbridas e sua relação com as lógicas de descrição e com revisão de crenças.

[1] http://jena.sourceforge.net/

Apêndice A

Graus de Recuperação em Bases de Crenças

No Capítulo 6 dissemos que a recuperação é o mais controverso dos postulados AGM para contração. Na teoria das bases de crenças esse postulado não é usado. No lugar dele são usados outros postulados para garantir a minimalidade da contração como a relevância e o core-retainment. Nesse capítulo trataremos de um campo intermediário entre a revisão em bases de crenças e a revisão em conjuntos de crenças. Trataremos da contração AGM *gerada* por uma contração em bases de crenças:

$$Cn(B) - \alpha = Cn(B - \alpha)$$

A recuperação não pode valer na contração gerada pela contração em bases que satisfaçam a inclusão para bases i.e.: $B - \alpha \subseteq B$

Proposição A.1. *[Han99] Seja $-$ uma contração em bases de crenças que satisfaz a inclusão e seja $B \neq \emptyset$, então $Cn(B) - \alpha = Cn(B - \alpha)$ não satisfaz o postulado da recuperação.*

Isso acontece porque a inclusão em bases de crenças é muito mais drástica do que a inclusão em conjuntos de crenças.

Neste capítulo mostraremos que relaxando a contração em bases de crenças geramos uma contração que satisfaz graus de recuperação. Esta contração é capaz de recuperar partes do conjunto de crenças original.

O postulado da inclusão em bases de crenças nem sempre é desejável. Os seguintes exemplos mostram situações em que a inclusão, da forma como foi definida, não é desejável.

Exemplo A.2: Suponha que nosso agente acredita que pássaros voam ($B = \{passaro \rightarrow voa\}$). Ao aprender que pinguins são pássaros e não voam podemos querer contrair a sentença $passaro \rightarrow voa$. Pela contração temos $B - passaro \rightarrow voa = \emptyset$. Porém, isso pode ser muito drástico. Uma solução possivelmente mais adequada seria $B - passaro \rightarrow voa = \{passaro \wedge \neg pinguim \rightarrow voa\}$. Ou seja, todo pássaros que não são pinguins voam. Evidente que existem casos em que de fato o mais adequado é remover $passaro \rightarrow voa$, mas essa segunda solução pode ser útil as vezes e é totalmente descartada pela inclusão.

Exemplo A.3: Seja uma base de crenças que contenha $s \wedge d$ que significa que Cleópatra tem um filho e uma filha, se queremos contrair s dessa base e aceitamos o postulado da inclusão somos obrigados a remover $s \wedge d$ e não podemos manter apenas d. Em outras palavras, se deixamos de acreditar que Cleópatra tem um filho, automaticamente deixamos de acreditar que ela tem uma filha também.

Esses exemplos mostram casos em que queremos manter partes das sentenças que devem ser removidas. Neste capítulo vamos mostrar duas construções para esse tipo de contração e algumas propriedades que elas satisfazem. Mostraremos que o enfraquecimento do postulado da inclusão leva ao fortalecimento do postulado da recuperação e vice-versa. Dessa forma, podemos ter graus de inclusão e recuperação.

Fermé e Hansson [FH99] trataram do caso dual em que queremos aceitar parte da entrada, utilizando uma função f tal que a nova revisão é dada por $K * f(\alpha)$.

A.1 Contração em Bases sem Inclusão

Os exemplos da seção anterior nos mostram que nem sempre o postulado da inclusão é desejado. Porém, não podemos simplesmente abandonar a inclusão. A inclusão garante que operações irracionais como $\{p\} - p = \{q\}$ não sejam permitidas. Não podemos simplesmente abandonar a inclusão, temos que enfraquecê-la para permitir a adição de alguns tipos específicos de fórmulas.

Uma forma de enfraquecer a inclusão foi proposta por Hansson em [Han89]:

(inclusão lógica) $B - \alpha \subseteq Cn(B)$

Operações de contração que satisfazem sucesso e inclusão lógica são chamadas de *pseudo-contrações*.

Da maneira análoga, podemos definir uma versão mais fraca da recuperação que chamaremos de *recuperação lógica*.

(recuperação lógica) $B \subseteq Cn(B - \alpha + \alpha)$

Nebel em [Neb89] propôs uma pseudo-contração que satisfaz a recuperação lógica:

Definição A.4. *[Neb89] Seja B uma base de crenças, α uma sentença γ uma função de seleção.*

$$B - \alpha = \begin{cases} B & \text{if } \alpha \in Cn(\emptyset) \\ \bigcap \gamma(B \perp \alpha) \cup \{\alpha \rightarrow \beta | \beta \in B\} & \text{caso contrário} \end{cases}$$

A idéia dessa construção é adicionar à contração partial meet algumas consequências das fórmulas da base original de forma a recuperar a base toda caso α seja novamente incorporado a base.

O problema com a definição acima é que não há nenhuma intuição por trás da adição de $\alpha \rightarrow \beta$ a não ser o fato de que isso permite recuperar a base. Se voltarmos ao exemplo A.3 a pseudo-contração de Nebel faria $\{p \wedge q\} - p = \{p \rightarrow p \wedge q\}$. Ou seja, ao contrair que Cleópatra tem um filho usando a construção de Nebel ainda não garantimos que Cleópatra tem uma filha, mas ao adicionar que ela tem um filho recuperamos a crença de que ela tem uma filha.

Apresentaremos uma generalização da contração acima em que permitimos a adição de algumas consequências da base.

Seja Cn^* um operador que gera algumas consequências de um conjunto de fórmulas ($Cn^*(B) \subseteq Cn(B)$). Por exemplo poderíamos ter $Cn^*(A) = \{\alpha \rightarrow \beta : \beta \in A\}$ ou $Cn^*(A) = \{\alpha : \alpha \wedge \beta \in A \text{ ou } \beta \wedge \alpha \in A\}$.

Definição A.5. *Seja γ uma função de seleção para B e B^* um conjunto de sentenças que contém B. Uma extensão de γ para B^* é uma função de seleção γ^* tal que para todo $Y \in \gamma^*(B^* \perp \alpha)$ existe um $X \in \gamma(B \perp \alpha)$ tal que $X \subseteq Y$.*

Definição A.6. *Seja B uma base de crenças, α uma sentença e γ uma função de seleção para B. A* pseudo-contração partial meet geral *$B - \alpha$ é dada por:*

$$B - \alpha = \begin{cases} B & \text{if } \alpha \in Cn(\emptyset) \\ \bigcap \gamma^*(B^* \perp \alpha) & \text{caso contrário} \end{cases}$$

onde $B^* = B \cup Cn^*(\{\beta | \beta \in B \setminus \bigcap \gamma(B \perp \alpha)\})$ e γ^* é uma extensão de γ.

Dependendo do operador Cn^* usado, esta construção satisfaz ou não a recuperação lógica. Porém, esta construção sempre satisfaz um postulado de minimalidade que chamaremos de *relevância lógica*:

(relevância lógica) Se $\beta \in B \setminus (B - \alpha)$, então existe B' tal que $B - \alpha \subseteq B' \subseteq Cn(B)$, $\alpha \notin Cn(B')$ e $\alpha \in Cn(B' \cup \{\beta\})$.

Proposição A.7. *A pseudo-contração definida acima satisfaz inclusão lógica, vacuidade, sucesso, a extensionalidade e a relevância lógica.*

Demonstração. A única parte mais problemática é provar que a relevância lógica é satisfeita. Para isso note que se $\beta \in B \setminus B - \alpha$, então existe $X \in \gamma^*(B^* \perp \alpha)$ tal que $\beta \notin X$. Como $B^* \subseteq Cn(B)$, podemos tomar $B' = X$. □

A.2 Adições Mínimas

Relembremos a construção de Nebel da definição A.4. Nessa construção $\{\alpha \to \beta | \beta \in B\}$ é adicionado ao resultado da contração partial meet para obter a recuperação lógica. É possível adicionarmos menos? Existem casos em que adicionando menos ainda assim a recuperação lógica é satisfeita?

Gostaríamos de garantir que apenas sentenças que são realmente necessárias para satisfazer a recuperação lógica sejam adicionadas na operação de contração. O postulado a seguir tenta capturar essa intuição:

(core-addition) Se $\beta \in (B - \alpha) \setminus B$, então existe $\beta' \in B \setminus \bigcap \gamma(B \perp \alpha)$ e $B' \subseteq B - \alpha$ tal que $\alpha \to \beta' \notin Cn(B')$ mas $\alpha \to \beta' \in Cn(B' \cup \{\beta\})$.

Esse postulado garante que se uma sentença foi adicionada na contração é por que ela é necessária para a recuperação. Core-addition é a contra-parte do core-retainment com respeito a mudança mínima: enquanto core-retainment previne perda desnecessária de informação, core-addition previne a adição desnecessária.

A construção a seguir usa a idéia de conjunto minimal que junto da contração partial meet recupera a base, mas não implica a sentença a ser contraída.

Definição A.8. *Seja Δ um subconjunto minimal de $Cn(B)$ tal que:*

1. *$\alpha \to \beta \in Cn(\bigcap \gamma(B \perp \alpha) \cup \Delta)$ para todo $\beta \in B \setminus (\bigcap \gamma(B \perp \alpha))$*
2. *Para todo $X \in B \perp \alpha$, temos $\alpha \notin Cn(X \cup \Delta)$*

Definimos Δ-partial-meet pseudo-contração *de B por α como: $B - \alpha = \bigcap \gamma(B \perp \alpha) \cup \Delta$*

Proposição A.9. *A pseudo-contração Δ-partial-meet satisfaz core-addition, sucesso, inclusão lógica e recuperação lógica.*

Essa construção é um caso particular da pseudo-contração partial meet geral. Basta considerar $Cn^*(B \setminus \bigcap \gamma(B \perp \alpha)) = \Delta$ e então $\bigcap \gamma^*(B^* \perp \alpha) = \bigcap \gamma(B \perp \alpha) \cup \Delta$. Ou seja, nesse caso ambas as operações coincidem:

Proposição A.10. *Seja Δ um subconjunto minimal de $Cn(B)$ tal que:*

1. *$\alpha \to \beta \in Cn(\bigcap \gamma(B \perp \alpha) \cup \Delta)$ para todo $\beta \in B \setminus (\bigcap \gamma(B \perp \alpha))$*
2. *Para todo $X \in B \perp \alpha$, temos $\alpha \notin Cn(X \cup \Delta)$*

e seja $Cn^*(B \setminus \bigcap \gamma(B \perp \alpha)) = \Delta$. Então $\bigcap \gamma^*(B^* \perp \alpha) = \bigcap \gamma(B \perp \alpha) \cup \Delta$.

Demonstração. Com $Cn^*(B \setminus \bigcap \gamma(B \perp \alpha)) = \Delta$ temos que $B^* = B \cup \Delta$. Pela definição da extensão γ^* sabemos que para todo elemento Y de $\gamma^*((B \cup \Delta) \perp \alpha)$ existe $X \in \gamma(B \perp \alpha)$ tal que $X \subseteq Y$. Pela definição de conjunto resíduo, como X é um subconjunto maximal de B que não implica α, e como pela definição de Δ sabemos que $\alpha \notin Cn(X \cup \Delta)$, temos que $Y = X \cup \Delta$. Logo, $\Delta \subseteq Y$ para todo $Y \in \gamma^*((B \cup \Delta) \perp \alpha)$ e, portanto, $\Delta \subseteq \bigcap \gamma^*((B \cup \Delta) \perp \alpha)$. Pela definição de γ^* concluímos que $\bigcap \gamma^*((B \cup \Delta) \perp \alpha) = \bigcap \gamma(B \perp \alpha) \cup \Delta$ □

A.3 Graus de Inclusão e Recuperação

Δ-Partial-meet pseudo-contração satisfaz core-adition e recuperação lógica. Porém, é possível definir uma construção para contração que satisfaça core-addition, mas não necessariamente satisfaça a recuperação lógica. Para tanto precisamos definir uma função que escolha quais as fórmulas que gostaríamos que fossem recuperáveis. Formalmente precisamos definir uma função f tal que $f(X) \subseteq X$.

Definição A.11. *Seja f uma função como a definida acima. Seja Δ_f um subconjunto minimal de $Cn(B)$ tal que:*

1. $\alpha \to \beta \in Cn(\bigcap \gamma(B \perp \alpha) \cup \Delta_f)$ *para todo* $\beta \in f(B \setminus (\bigcap \gamma(B \perp \alpha)))$
2. *Para todo* $X \in B \perp \alpha$, *temos* $\alpha \notin Cn(X \cup \Delta_f)$.

Definimos a Δ_f-partial-meet pseudo-contração de B por α como $B - \alpha = \bigcap \gamma(B \perp \alpha) \cup \Delta_f$.

Essa construção torna clara a relação entre os diferentes graus de recuperação. Em um extremo, se f escolhe todo o conjunto, temos a recuperação lógica; no outro extremo, se f seleciona o conjunto vazio, a contração satisfaz a inclusão para bases, logo, nenhuma sentença é adicionada. Entre esses extremos existe todo um universo de contrações que recuperam partes da base. Da mesma forma que a construção da seção anterior essa construção é um caso particular da pseudo-contração partial meet geral.

A.4 Conclusão

Neste capítulo estudamos uma área entre a área de revisão em bases e revisão em conjuntos de crenças. Estudamos a contração gerada por uma base. Em particular estudamos a relação entre a inclusão e a recuperação em contrações geradas por bases.

A recuperação na teoria AGM é o mais controverso dos postulados para contração. Na teoria de bases de crenças esse postulado não se aplica.

A inclusão, indiscutível na teoria AGM, é um postulado controverso na teoria de bases de crenças. A presença da inclusão em bases impede que tratemos de exceções (exemplo A.2) e não é muito intuitiva em outros casos (exemplo A.3). Ao enfraquecermos o postulado da inclusão em bases de crenças podemos fortalecer a recuperação obtendo graus de recuperação.

Mostramos nessa seção uma construção para contração em bases de crenças que gera uma contração em conjuntos de crenças que satisfaz graus de recuperação. Ou seja, essa contração é capaz de recuperar parte do conjunto de crenças original.

Apêndice B

Provas

B.1 Provas do Capítulo 5

B.1.1 Relações entre propriedades lógicas

Nas demonstrações a seguir usaremos o seguinte abuso de notação: Usaremos $\neg A$ para representar um conjunto qualquer que seja negação clássica de A, ou seja, um conjunto Y tal que $Cn(Y) \cap Cn(A) = Cn(\emptyset)$ e $Cn(Y \cup A) = \mathscr{L}$.

Proposição 5.12 Se uma lógica $\langle \mathscr{L}, Cn \rangle$ é supra-clássica e satisfaz dedução então $\langle \mathscr{L}, Cn \rangle$ é booleana.

Demonstração. A *distributividade* segue da observação 1.15 de [Han99].

Para mostrar que a lógica é *fechada por negação* considere um conjunto qualquer $X \in 2^{\mathscr{L}}$. Se X é finitamente representável então $Cn(X) = Cn(X')$ para algum X' finito. Como X' é finito podemos enumerar seus elementos $X' = \{\beta_1, \ldots, \beta_n\}$ e temos que $Cn(X') = Cn(\{\beta_1, \ldots, \beta_n\}) = Cn(\beta_1 \wedge \cdots \wedge \beta_n)$. Seja $\alpha = \beta_1 \wedge \cdots \wedge \beta_n$, temos que mostrar que α possui uma negação clássica, pois assim, X possuirá negação clássica visto que $Cn(X) = Cn(X') = Cn(\alpha)$.

Para qualquer $\beta \in \mathscr{L}$ temos que $\beta \in C_{LPC}(\{\alpha, \neg\alpha\})$ onde C_{LPC} é o operador de consequência da lógica proposicional clássica. Como a lógica $\langle \mathscr{L}, Cn \rangle$ é supra-clássica temos que $\beta \in Cn(\{\alpha, \neg\alpha\})$. Para completar a prova vamos mostrar agora que $Cn(\alpha) \cap Cn(\neg\alpha) \subseteq Cn(\emptyset)$. Seja $\beta \in Cn(\alpha) \cap Cn(\neg\alpha)$ então pela dedução temos que $\alpha \to \beta \in Cn(\emptyset)$ e $\neg\alpha \to \beta \in Cn(\emptyset)$. Pela supra-classicalidade temos que $\beta \in Cn(\emptyset)$. □

Proposição 5.14: Lógicas decomponíveis que satisfaçam a condição da cadeia descendente são fechadas por negação.

Demonstração. Se $A = \mathscr{L}$ então $Cn(\emptyset)$ é uma negação clássica de A. Seja $A \subset \mathscr{L}$, pela decomponibilidade temos que $\mathcal{A}^-(\mathscr{L}) \neq \emptyset$, ou seja, existe algum X_0 tal que $Cn(\emptyset) \subset Cn(X_0) \subset \mathscr{L}$ e $Cn(X_0 \cup A) = \mathscr{L}$. Como $\langle \mathscr{L}, Cn \rangle$ satisfaz a condição da cadeia descendente, deve haver uma cadeia começando em X_0 que possui um último elemento $X_n \in A^-(\mathscr{L})$. Assim, $X_n \subseteq X_0$ e não existe $X_i \in A^-(\mathscr{L})$ tal que $Cn(X_i) \subset Cn(X_n)$ (a figura B.1 ilustra essa parte). Vamos provar que X_n é a negação de A. Como $X_n \in A^-(\mathscr{L})$ temos que $Cn(X_n \cup A) = \mathscr{L}$. Agora suponha que $Cn(X_n) \cap Cn(A) \neq Cn(\emptyset)$, como a lógica é decomponível, existe $Y \in (Cn(X_n) \cap Cn(A))^-(Cn(X_n))$.

Nesse caso temos que $X_n = Cn(Y \cup (Cn(A) \cap Cn(X_n))) \subseteq Cn(Y \cup A)$. Porém, como $A \subseteq Cn(Y \cup A)$, temos que $Cn(X_n \cup A) \subseteq Cn(Y \cup A)$. Logo $Cn(Y \cup A) = \mathscr{L}$.

Nesse caso $Cn(Y) \subset Cn(X_n)$ e $Cn(Y \cup A) = \mathscr{L}$. Segue que X_n não é minimal contrariando a definição. Concluímos que $Cn(X_n) \cap Cn(A) = Cn(\emptyset)$. □

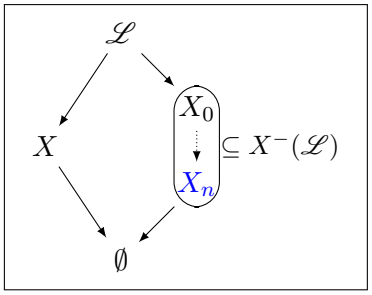


Figura B.1: A negação em lógicas finitas implica na decomponibilidade.

Proposição 5.15: Seja $\langle \mathscr{L}, Cn\rangle$ uma lógica distributiva. Se $Cn(A) \cap Cn(A') = Cn(\emptyset)$ então A' é uma negação de A que satisfaz a não contravenção.

Demonstração. Suponha que $A' \subseteq Cn(B \cup A)$ para $B \in 2^{\mathscr{L}}$. Temos que mostrar que $A' \subseteq Cn(B)$.

Como a lógica é tarskiana temos $Cn(A') \subseteq Cn(B \cup A)$ por monotonicidade e idempotência. Pela monotonicidade temos $Cn(A') \subseteq Cn(B \cup A')$. Dessas duas sentenças temos $Cn(A') \subseteq Cn(B \cup A) \cap Cn(B \cup A')$. Pela distributividade inferimos que $Cn(A') \subseteq Cn(B \cup (Cn(A) \cap Cn(A')))$. Por hipótese $Cn(A) \cap Cn(A') = Cn(\emptyset)$. Logo, $Cn(A') \subseteq Cn(B \cup Cn(\emptyset)) = Cn(B)$. □

B.1.2 Lógica de Horn

Proposição 5.18 A lógica de Horn $\langle \mathscr{L}, Cn\rangle$ não é decomponível.

Demonstração. Vamos usar o lema 5.26: se existe algum conjunto $K \in 2^{\mathscr{L}}$ tal que $Cn(\emptyset) \subset Cn\{x \in \mathscr{L} : Cn(x) \subset Cn(K)\} \subset Cn(K)$ então $\langle \mathscr{L}, Cn\rangle$ não é decomponível.

Seja $K = \{b_0\}$ tal que $b_0 \in \mathbf{P}$ e seja $Y = Cn\{x \in \mathscr{L} : Cn(x) \subset Cn(K)\}$. Então $Cn(\emptyset) \subset Cn(Y)$ porque pelo menos um $b_i \to b_0 \in Y$ para cada $i > 0$.

Agora temos que provar que $Cn(Y) \subset Cn(b_0)$. $\beta \in Y$ é uma conjunção de cláusulas de Horn. Cada cláusulas em β tem exatamente um literal não negado b_0, caso contrário $\beta \notin Cn(b_0)$. Todas cláusulas tem que ter pelo menos um literal não negado, senão $Cn(\beta) = Cn(b_0)$. Logo temos que, a atribuição **falso** para todo $b_i \in \mathbf{P}$ satisfaz β, mas não satisfaz b_0. Logo, $Cn(Y) \subset Cn(b_0)$. □

Proposição 5.19 A lógica de Horn é compacta.

Demonstração. Pela definição que demos de lógica de Horn temos que se $\alpha \in Cn_{Horn}(X)$ então $\alpha \in Cn_{LPC}(X)$. Como a LPC é compacta, existe um conjunto finito $X' \subseteq X$ tal que $\alpha \in Cn_{LPC}(X')$ e como $X' \subseteq X$ vale que $\alpha \in Cn_{Horn}(X')$. □

B.1.3 Lógica intuicionista

Proposição 5.21 A lógica intuicionista não é decomponível.

Demonstração. Seja $K = \{\neg p\}$ e $A = \{\neg p \vee p\}$. Mostraremos que: 1) $Cn(\emptyset) \subset Cn(A) \subset Cn(K)$ e 2) Não existe X tal que $Cn(X) \subset Cn(K)$ e $Cn(X \cup A) = K$

1) Usando dedução natural temos que $p \vee \neg p \in Cn(\neg p)$.

Para mostrar que $Cn(p \vee \neg p) \neq Cn(\neg p)$ vamos usar a semântica de Kripke da lógica intuicionista. Temos que mostrar um modelo $\langle \mathbf{V}, \mathbf{W}, \mathbf{R}\rangle$ tal que $M \vDash p \vee \neg p$, mas $M \nvDash \neg p$. O seguinte modelo serve para isso: $W = \{w\}$, $R = \{(w, w)\}$ e $V(p) = \{w\}$.

Para mostrar que $p \vee \neg p \notin Cn(\emptyset)$, temos que mostrar um $\langle \mathbf{V}, \mathbf{W}, \mathbf{R}\rangle$ tal que $M \nvDash p \vee \neg p$. Considere o seguinte modelo:

$$\begin{aligned} W &= \{w_0, w_1\} \\ R &= \{(w_0, w_0), (w_0, w_1), (w_1, w_1)\} \\ V(p) &= \{w_1\} \end{aligned}$$

2) Vamos mostrar que para qualquer X tal que $Cn(X) \subset Cn(\neg p)$ existe um modelo M tal que $M \vDash X, p \vee \neg p$, mas $M \nvDash \neg p$.

Tome X tal que $Cn(X) \subset Cn(\neg p)$. Como $\neg p \notin Cn(X)$ existe um modelo $M'\langle V', W', R'\rangle$ de X ($M' \vDash X$) tal que existe um $w \in W'$ com $(w_0, w) \in R'$ e $w \in V'(p)$. Agora, considere um $\langle \mathbf{V}, \mathbf{W}, \mathbf{R}\rangle$ tal que $W = \{w \in W' : w \in V'(p)\}$, $V(q) = V'(q) \cap W$ para todo q e $R = \{(w_1, w_2) \in R' : w_1, w_2 \in W\}$. Então, temos que, $M \vDash X$ e $M \vDash p$. Segue que $M \vDash p \vee \neg p, X$, mas $M \nvDash \neg p$. □

B.1.4 Lógicas de descrição

Proposição 5.29 Toda lógica de descrição que é equivalente a um subconjunto da lógica de primeira ordem é compacta.

Demonstração. Considere $\alpha \in Cn(X)$ tal que X é um conjunto de sentenças em uma LD e α é uma sentença nessa lógica. Por hipótese é possível traduzir X e α em um conjunto de sentenças X_{FOL} e em uma sentença α_{FOL} na lógica de primeira ordem. Como a lógica de primeira ordem é compacta, existe uma subconjunto X'_{FOL} de X_{FOL} que é finito e tal que $\alpha_{FOL} \in Cn(X'_{FOL})$. Como $X'_{FOL} \subseteq X_{FOL}$ é possível traduzir X'_{FOL} de volta para a lógica de descrição original X'. Esse X' é finito e $\alpha \in Cn(X')$. Logo, a LD é compacta. □

Proposição 5.30 Considere uma LD $\langle \mathscr{L}, Cn\rangle$ tal que para qualquer sentença $\alpha \in \mathscr{L}$ existe $\alpha' \in \mathscr{L}$ tal que $Cn(\alpha) = Cn(\alpha')$ e α' tem a forma $\top \sqsubseteq C$ para algum C. Então $\langle \mathscr{L}, Cn\rangle$ é distributiva.

Demonstração. Queremos mostrar que para qualquer $X, Y, Z \in 2^{\mathscr{L}}$ vale que $Cn(X \cup (Cn(Y) \cap Cn(Z))) = Cn(X \cup Y) \cup Cn(X \cup Z)$.

Suponha sem perda de generalidade que $X = \{\top \sqsubseteq A_1, \ldots, \top \sqsubseteq A_n\}$, $Y = \{\top \sqsubseteq B_1, \ldots, \top \sqsubseteq B_{n'}\}$ e $Z = \{\top \sqsubseteq C_1, \ldots, \top \sqsubseteq C_{n''}\}$. Se $\top \sqsubseteq D \in Cn(X \cup Y) \cap Cn(X \cup Z)$ então traduzindo para a lógica de primeira ordem temos $\vDash \forall x A_1(x) \wedge \cdots \wedge \forall x A_n(x) \wedge \forall x B_1(x) \wedge \ldots \forall x B_{n'}(x) \rightarrow \forall x D(x)$ e $\vDash \forall x A_1(x) \wedge \cdots \wedge \forall x A_n(x) \wedge \forall x C_1(x) \wedge \ldots \forall x C_{n'}(x) \rightarrow \forall x D(x)$. Segue que $\vDash \neg(\forall A_1(x) \wedge \cdots \wedge \forall x A_n(x))$ ou $\vDash (\forall x C_1(x) \wedge \ldots \forall x C_{n''}(x) \rightarrow \forall x D(x)) \wedge (\forall x B_1(x) \wedge \ldots \forall x B_{n'}(x) \rightarrow \forall x D(x))$.

No primeiro caso $Cn(X) = \mathscr{L}$ e, pela monotonicidade, $\top \sqsubseteq D \in Cn(X \cup (Cn(Y) \cap Cn(Z)))$.

No segundo caso $\top \sqsubseteq D \in Cn(Y) \cap Cn(Z)$. Pela monotonicidade segue que $\top \sqsubseteq D \in Cn(X \cup (Cn(Y) \cap Cn(Z)))$.

Logo, se $\top \sqsubseteq D \in Cn(X \cup Y) \cap Cn(X \cup Z)$, então $\top \sqsubseteq D \in Cn(X \cup (Cn(Y) \cap Cn(Z)))$. □

Lema 5.34 Sejam A e B dois conceitos tais que $\top \sqsubseteq A$ e $\top \sqsubseteq B$ não são tautologias e suponha que exista um papel R não relacionado com nenhum papel que ocorre em A ou em B. Então, $Cn(\top \sqsubseteq A) \cap Cn(\top \sqsubseteq B) \neq Cn(\emptyset)$.

Demonstração. Como $\top \sqsubseteq A$ não é uma tautologia, existe pelo menos uma interpretação $\mathcal{I}$ tal que $A^{\mathcal{I}} \neq \Delta^{\mathcal{I}}$. Suponha que para toda interpretação $\mathcal{I}$, se $A^{\mathcal{I}} \neq \Delta^{\mathcal{I}}$ então $B^{\mathcal{I}} = \Delta^{\mathcal{I}}$. Segue que $A \sqsubseteq B \in Cn(\emptyset)$ e $\top \sqsubseteq B \in Cn(\top \sqsubseteq A) \cap Cn(\top \sqsubseteq B)$. Como $\top \sqsubseteq B \notin Cn(\emptyset)$, o resultado está provado.

Agora suponha que existe uma interpretação $\mathcal{I}$ tal que $A^{\mathcal{I}} \neq \Delta^{\mathcal{I}}$ e $B^{\mathcal{I}} \neq \Delta^{\mathcal{I}}$. Mostraremos que se escolhermos um $R \in ROL$ que não foi usado nem em A nem B então $\top \sqsubseteq A \sqcup \forall R.B \notin Cn(\emptyset)$. Se mostrarmos isso, como $\top \sqsubseteq A \sqcup \forall R.B \in Cn(\top \sqsubseteq A) \cap Cn(\top \sqsubseteq B)$, teremos provado o resultado.

Considere uma interpretação $\mathcal{I}$ tal que $A^{\mathcal{I}} \neq \Delta^{\mathcal{I}}$ e $B^{\mathcal{I}} \neq \Delta^{\mathcal{I}}$. Existe então $a, b \in \Delta^{\mathcal{I}}$ tais que $a \notin A^{\mathcal{I}}$ e $b \notin B^{\mathcal{I}}$. Seja $(a, b) \in R^{\mathcal{I}}$, então $a \notin \forall R.B$. Segue que $a \notin A^{\mathcal{I}} \cup (\forall R.B)^{\mathcal{I}}$. Logo, $\top \sqsubseteq A \sqcup \forall R.B \notin Cn(\emptyset)$. □

Lema B.1. *Seja B uma negação de $Cn(\emptyset)$, então $Cn(B) = \mathscr{L}$.*

Demonstração. Seja B uma negação de $Cn(\emptyset)$ então $Cn(B) \cup Cn(\emptyset) = \mathscr{L}$, mas como $Cn(\emptyset) \subseteq Cn(B)$ então $Cn(B) = \mathscr{L}$. □

Proposição 5.33 Considere um LD $\langle \mathscr{L}, Cn \rangle$ tal que para qualquer sentença $\alpha \in \mathscr{L}$ existe $\alpha' \in \mathscr{L}$ tal que $Cn(\alpha) = Cn(\alpha')$ e α' tem a forma $\top \sqsubseteq C$. Se a assinatura de $\langle \mathscr{L}, Cn \rangle$ possui uma infinidade de papéis não relacionados, então $\langle \mathscr{L}, Cn \rangle$ não é fechada por negação.

Demonstração. Seja A um conceito tal que $Cn(\top \sqsubseteq A) \neq \mathscr{L}$. Suponha por absurdo que $\top \sqsubseteq B$ é uma negação de $\top \sqsubseteq A$. Pelo lema B.1, se B é uma negação de A então $Cn(\top \sqsubseteq B) \neq Cn(\emptyset)$. Logo, tanto $\top \sqsubseteq A$ quanto $\top \sqsubseteq B$ não são tautologias. Por hipótese deve haver um papel R que não se relaciona a nenhum papel que ocorre em A ou em B, pois apenas um número finito de papéis devem ocorrer em um conceito. Logo, pelo lema 5.34 temos que $Cn(\top \sqsubseteq A) \cap Cn(\top \sqsubseteq B) \neq Cn(\emptyset)$, mas isso contraria o fato de $\top \sqsubseteq B$ ser uma negação de $\top \sqsubseteq A$. Concluímos que $\top \sqsubseteq A$ não possui negação. □

B.2 Provas do Capítulo 6

Proposição 6.5 Em lógicas booleanas, na presença dos demais postulados AGM para contração, a relevância é equivalente à recuperação.

Demonstração. **(relevância ⇒ recuperação)**

Vamos mostrar que se $\beta \notin K - A + A$ então $\beta \notin K$.

Suponha por absurdo que $X = Cn(\neg A) \cap Cn(\beta) \subseteq K$. Pela distributividade temos que $X + A = (\neg A + A) \cap (A + \beta) = A + \beta$. Segue que $\beta \in X + A$, mas como $\beta \notin K - A + A$, então temos que $X \nsubseteq K - A$. Segue que existe algum $x \in X$ tal que $x \in K$, mas $x \notin K - A$. Pela *relevância* existe $K' \subseteq K$ tal que $A \nsubseteq Cn(K')$, mas $A \subseteq K' + x$. Segue que $A \subseteq K' + X$. Pela distributividade $A \subseteq K' + X = (K' + \neg A) \cap (K' + \beta)$, então $A \subseteq K' + \neg A$. Nesse caso, $A \subseteq (K' + \neg A) \cap (K' + A) = Cn(K' \cap (A + \neg A)) = Cn(K' \cap \mathscr{L}) = Cn(K')$ que é uma contradição. Logo, $X \nsubseteq Cn(K)$ e $Cn(\beta) \nsubseteq Cn(K)$ ou de forma equivalente $\beta \notin Cn(K)$

(recuperação ⇒ relevância)

Para provar que satisfaz a relevância temos que ter que $\beta \in K$ e $\beta \notin K - A$ implicam que existe K' tal que: 1) $K - A \subseteq K' \subseteq K$, 2) $A \subseteq K' + \beta$ e 3) $A \nsubseteq Cn(K')$

Vamos mostrar que isso é válido para $K' = K - A + (Cn(A) \cap Cn(\neg\beta))$.

Para o caso em que $A \nsubseteq K$ temos pela *vacuidade* que $K - A = K$. Segue que não existe β satisfazendo ambas condições. Logo, a relevância é trivialmente satisfeita. Considere, então, $A \subseteq K$:

1) $K - A \subseteq K'$ segue da construção. Para ver que $K' \subseteq K$, note que por *inclusão* $K - A \subseteq K$. Segue que $K - A \cup (Cn(A) \cap Cn(\neg\beta)) \subseteq K + A = K$, pois $A \subseteq K$

2) $(K - A \cup (Cn(A) \cap Cn(\neg\beta))) + \beta = K - A \cup (A + \beta)) \cap (\neg\beta + \beta) = K - A \cup (A + \beta)$. Logo $A \subseteq K' + \beta$

3) Suponha que $A \subseteq K - A + (Cn(A) \cap Cn(\neg\beta)) \subseteq K - A + \neg\beta$. Segue que $Cn(A) \cap Cn(\beta) \subseteq K - A + \neg\beta \cap (K - A + \beta)$. Pela distributividade temos que $Cn(A) \cap Cn(\beta) \subseteq K - A + (Cn(\beta) \cap Cn(\neg\beta))) = K - A$.

Como $\beta \in Cn(K)$ pelo lema B.2 $Cn(\neg A) \cap Cn(\beta) \subseteq K - A$. Como $Cn(\neg A) \cap Cn(\beta) \subseteq K - A$ e $Cn(A) \cap Cn(\beta) \in K - A$, temos que $(Cn(\neg A) \cap Cn(\beta)) \cup (Cn(A) \cap Cn(\beta)) = (Cn(A) \cap Cn(\neg A)) + \beta = Cn(\beta) \subseteq K - A$. Porém, isso contradiz a definição de β. Logo, $A \nsubseteq K'$.

□

Lema B.2. *Seja $\langle \mathscr{L}, Cn\rangle$ uma lógica booleana. Se $\beta \in Cn(K)$ e $K - A$ satisfaz a recuperação então $Cn(\neg A) \cap Cn(\beta) \subseteq K - A$*

Demonstração. $\beta \in K - A + A = K$ (recuperação). Segue que $Cn(\neg A) \cap Cn(\beta) \subseteq (K - A + A) \cap (K - A + \neg A)$. Pela distributividade $Cn(\neg A) \cap Cn(\beta) \subseteq K - A + (Cn(A) \cap Cn(\neg A)) = K - A$ □

Teorema de representação 6.8 Seja $\langle \mathscr{L}, Cn\rangle$ uma lógica compacta e seja A finitamente representável, então $K - A$ satisfaz a relevância e os demais postulados AGM sse $K - A = \bigcap \gamma(K \perp A)$ para alguma função de seleção γ.

Demonstração. **(construção ⇒ postulados)** *Inclusão* e *fecho* seguem da definição de γ. Para *vacuidade* note que se $A \nsubseteq K$ então $K \perp A = \{K\}$. *Extensionalidade* segue do fato que se $Cn(A) = Cn(B)$ então $K \perp A = K \perp B$. *Sucesso* segue do lema B.6 . Finalmente, para provar a *relevância*, no caso limite$A \subseteq Cn(\emptyset)$, pelo lema B.6, $K \perp A = \emptyset$ e $K - A = K$, então o postulado é trivialmente satisfeito. Se $K \perp A \neq \emptyset$, tome qualquer $K' \in \gamma(K \perp A)$ e qualquer $\beta \in K$ tal que $\beta \notin K - A$. Pela definição de $K - A$ e de conjunto resto, temos que $K - A \subseteq K' \subseteq K$ e que $A \nsubseteq Cn(K')$, mas $A \subseteq Cn(K' \cup \{\beta\})$.

(postulados ⇒ construção) Considere um operador $-$ satisfazendo os seis postulados. Defina a função de seleção γ como $\gamma(K \perp A) = \{K\}$ quando $K \perp A = \emptyset$ e $\gamma(K \perp A) = \{X \in K \perp \alpha : K - \alpha \subseteq X\}$ caso contrário.

Primeiro temos que provar que γ é uma função. Se $A \nsubseteq K$ então pelo lema B.7 $B \nsubseteq K$ e pela *vacuidade* temos que $K - A = K - B = \{K\}$. Logo $\gamma(K \perp A) = \gamma(K \perp B)$.

Se $A \subseteq K$ então $Cn(A) \subseteq Cn(K)$ e pelo lema B.7 temos que $B \subseteq Cn(A)$. Analogamente, $A \subseteq Cn(B)$. Segue que $Cn(A) = Cn(B)$ e então, pela *extensionalidade* $K - A = K - B$. Logo $\gamma(K \perp A) = \gamma(K \perp B)$.

Além disso, pelo *sucesso*, *inclusão* e pela propriedade do limite superior (lema B.5), segue que se $K \perp A \neq \emptyset$, então $\gamma(K \perp A) \neq \emptyset$, i.e., γ é uma função de seleção.

Para mostrar que $K - A = \bigcap \gamma(K \perp A)$ dividiremos o problema em dois. Primeiro suponha que $A \subseteq Cn(\emptyset)$ segue da definição que $K - A = \bigcap \gamma(K \perp A) = \{K\}$. Agora suponha que $A \nsubseteq Cn(\emptyset)$, temos pela definição de γ que $K - A \subseteq \bigcap \gamma(K \perp A)$. Para provar $\bigcap \gamma(K \perp A) \subseteq K - A$ temos que provar que se $\beta \notin K - A$ então $\beta \notin \bigcap \gamma(K \perp A)$. Se $\beta \notin K$ então (pela definição) $\beta \notin \bigcap \gamma(K \perp A)$, assim suponha que $\beta \in K$. Pela *relevância*, temos que existe K' tal que $K - A \subseteq K' \subseteq K$, $A \nsubseteq Cn(K')$, but $A \subseteq Cn(K' \cup \{\beta\})$. Pelo lema B.5 (propriedade generalizada da limite superior) que existe K'' tal que $K' \subseteq K'' \in K \perp A$, mas é fácil ver que $\beta \notin K''$ (caso contrário teríamos $A \subseteq Cn(K'')$). Segue de $K - A \subseteq K' \subseteq K''$ que $K'' \in \gamma(K \perp A)$ e concluímos que $\beta \notin \bigcap \gamma(K \perp A)$ □

Lema B.3 (propriedade do limite superior)**.** *[AM81] Toda lógica compacta satisfaz a seguinte propriedade: para todo $K, A, X \subseteq \mathscr{L}$ tal que $X \subseteq K$ e $A \nsubseteq Cn(X)$, existe algum $X' \subseteq \mathscr{L}$ tal que $X \subseteq X' \in K \perp A$*

Lema B.4. *Seja $A = \{a_0, a_1, \dots\}$ então para todo $K \subseteq \mathscr{L}$ temos que $\{X \in K \perp a_i : \forall a \in A(\nexists X' \in K \perp a \text{ tal que. } X \subset X')\} \subseteq K \perp A$.*

Demonstração. Temos que provar 1) $A \nsubseteq Cn(X)$, 2) $X \subseteq K$ e 3) Para todo X' tal que $X \subset X' \subseteq K$, $A \subseteq Cn(X')$:

1) Como $X \in K \perp a_i$ para algum a_i então $a_i \notin Cn(X)$. Logo, $A \nsubseteq Cn(X)$.

2) Como $X \in K \perp a_i$ para algum a_i então $X \subseteq K$.

3) Suponha que existe X' tal que $X \subset X' \subseteq K$ e $A \nsubseteq Cn(X')$ então pegue $a_k \in A \setminus Cn(X')$. Pelo lema B.3 existe X'' tal que $X' \subseteq X''$ e $X'' \in K \perp a_k$, mas nesse caso $X \subseteq X' \subset X'' \in K \perp a_k$ contradizendo a definição. □

Lema B.5 (propriedade generalizada do limite superior)**.** *Se uma $\langle \mathscr{L}, Cn\rangle$ satisfaz a propriedade do limite superior para fórmulas então ela também satisfaz para conjuntos finitamente representáveis.*

Demonstração. Como $A \nsubseteq Cn(X)$ existe $A' \subseteq A$ tal que $a_i \notin Cn(X)$ para cada $a_i \in A'$. Pela propriedade do limite superior, para cada $a_i \in A'$ existe um X_i' tal que $X \subseteq X_i'$ e $X_i' \in K \perp a_i$. Como $A' \subseteq A$ é finito tem de existir pelo menos um X_i' que não está incluído em nenhum dos outros. Logo, pelo lema B.4, esse $X_i' \in \{X \in K \perp a_i : \forall a \in A(\nexists X' \in K \perp a \text{ s.t. } X \subset X')\} \subseteq K \perp A$. □

Lema B.6. *Seja $\langle \mathscr{L}, Cn\rangle$ be uma lógica que satisfaz a propriedade do limite superior. Seja $K \subseteq \mathscr{L}$ um conjunto de crenças e seja $A \subseteq \mathscr{L}$ finitamente representável. $K \perp A = \emptyset$ sse $A \subseteq Cn(\emptyset)$*

Demonstração. Segue direto do lema B.5 com $X = \emptyset$. □

Lema B.7. *Se uma lógica $\langle \mathscr{L}, Cn\rangle$ satisfaz a propriedade do limite superior então $\langle \mathscr{L}, Cn\rangle$ também satisfaz o seguinte: $K \perp A = K \perp B$ implica que para todo $K' \subseteq K$, $A \subseteq Cn(K')$ sse $B \subseteq Cn(K')$*

Demonstração. Se $K \perp A = K \perp B$ então se $A \nsubseteq Cn(K')$, mas $B \subseteq Cn(K')$, pelo lema B.5, existe X tal que $K' \subseteq X \in K \perp A$, but $X \notin K \perp B$. □

Lema B.8. *Se $\langle \mathscr{L}, Cn\rangle$ é booleana então $K + A = \mathscr{L}$ implica $\neg A \subseteq Cn(K)$*

Demonstração. Como a lógica é fechada por negação $\neg A \subseteq \mathscr{L}$. Segue que $\neg A \subseteq K + A$ e $\neg A \subseteq K + \neg A$. Temos, então, que $\neg A \subseteq K + A \cap K + \neg A$. Pela distributividade, $\neg A \subseteq K + (Cn(A) \cap Cn(\neg A)) = K + Cn(\emptyset) = Cn(K)$. □

B.3 Provas do Capítulo 7

Proposição 7.3 Seja $\langle \mathscr{L}, Cn\rangle$ uma lógica tarskiana e compacta, $K *_\gamma A$ satisfaz os postulados AGM básicos para revisão.

Demonstração. Fecho, *sucesso* e *inclusão* seguem diretamente da construção. Se $K+A$ é consistente então $K \downarrow A = \{K\}$ e, portanto, a *vacuidade* é satisfeita. Se $Cn(A) = Cn(B)$ então para todo X, $X \cup A$ é inconsistente sse $X \cup B$ também é inconsistente, logo, a *extensionalidade* é satisfeita. Finalmente, para provar a *consistência*, suponha que A é consistente e que $\bigcap \gamma(K \downarrow A) + A$ é inconsistente. Como $\bigcap \gamma(K \downarrow A) \subseteq X \in K \downarrow A$, pela monotonicidade $X \cup A$ é inconsistente, o que contradiz a definição. □

Lema B.9. *(Teorema 8.5 de [Del08]) $K \downarrow A = K \downarrow B$ sse para todo $X \subseteq K$, $X \cup A$ é inconsistente sse $X \cup B$ é inconsistente.*

Proposição 7.4 $K *_\gamma A = \bigcap \gamma(K \downarrow A) + A$ satisfaz a uniformidade e a relevância.

Demonstração. Primeiro provaremos que a construção satisfaz a *relevância*. seja $\beta \in K \setminus K *_\gamma A$. Existe $X \in \gamma(K \downarrow A)$ tal que $\beta \notin X + A$. Também sabemos que para qualquer $X' \in \gamma(K \downarrow A)$, vale que $\bigcap \gamma(K \downarrow A) + A \subseteq X' + A$ e portanto, $K \cap (\bigcap \gamma(K \downarrow A) + A) \subseteq K \cap (X' + A)$. Isso vale em particular para o X acima. Tome então $K' = K \cap (X + A)$.

Para a *uniformidade*, note que se vale que para todo $K' \subseteq K$, $K' \cup A$ é inconsistente sse $K' \cup B$ é inconsistente, então pelo lema B.9, $K \downarrow A = K \downarrow B$, e logo, $\bigcap \gamma(K \downarrow A) = \bigcap \gamma(K \downarrow B)$. □

Proposição 7.5 Considere que a lógica subjacente é distributiva e monotônica. Se um operador $*$ satisfaz:

1. sucesso, inclusão e relevância então $*$ satisfaz a vacuidade.

2. sucesso, inclusão e uniformidade então $*$ satisfaz a extensionalidade.

Demonstração. 1. Se $K+A$ é consistente então não existe $K' \subseteq K$ tal que $K'+A$ é inconsistente. Logo, pela *relevância* não existe $\beta \in K \setminus K * A$. Portanto $K \subseteq K * A$. Pelo *sucesso* temos que $K + A \subseteq K * A$ e pela *inclusão* $K * A \subseteq K + A$.

2. Se $Cn(A) = Cn(B)$ para todo $K' \subseteq K$ temos que $K' + A$ é inconsistente sse $K' + B$ é inconsistente. Logo, pela *uniformidade*, $K \cap K * A = K \cap K * B$. Como $Cn(A) = Cn(B)$, segue que $(K \cap K * A) + A = (K \cap K * B) + B$. Pela *distributividade* $K + A \cap K * A + A = K + B \cap K * B + B$. Por *sucesso* e *inclusão* temos $K * A = K * B$.

□

Lema B.10. *Em qualquer lógica monotônica e compacta se $X \subseteq K$ e $X \cup A$ é consistente então existe X' tal que $X \subseteq X' \in K \downarrow A$.*

Demonstração. Essa é uma generalização direta da *propriedade do limite superior* [AGM85]. □

Lema B.11 (Teorema 8.5 de [Del08])**.** *$K \downarrow A = K \downarrow B$ sse para todo $X \subseteq K$, $X \cup A$ é inconsistente sse $X \cup B$ é inconsistente.*

Lema B.12. *Se a lógica subjacente é distributiva e '$*$' satisfaz sucesso, inclusão e consistência, então $(K \cap K * A) + A = K + A \cap K * A + A = K * A$*

Demonstração. $(K \cap K * A) + A = Cn((K \cap K * A) \cup A)$ =(pela distributividade) $Cn(K \cup A) \cap Cn(K * A \cup A) = K + A \cap K * A + A =$ (por sucesso e consistência) $K + A \cap K * A =$ (pela inclusão)$K * A$ □

Teorema de representação 7.6 Em qualquer lógica distributiva e compacta que satisfaça a explosão inconsistente, uma operação $*$ é uma revisão sem negação $\bigcap \gamma(K \perp A) + A$ para alguma função de seleção γ sse $*$ satisfaz fecho, sucesso, inclusão, consistência, relevância e uniformidade.

Demonstração. **(construção ⇒ postulados)**

Pela proposição 7.3, todos postulados AGM para revisão são satisfeitos pela construção.

Pela *relevância*, seja $\beta \in K \setminus K *_\gamma A$. Então existe $X \in \gamma(K \downarrow A)$ tal que $\beta \notin X + A$. Também sabemos que para todo $X' \in \gamma(K \downarrow A)$, vale que $\bigcap \gamma(K \downarrow A) + A \subseteq X' + A$. Logo, $K \cap (\bigcap \gamma(K \downarrow A) + A) \subseteq K \cap (X' + A)$. Isso vale em particular para o X como acima. Tome $K' = K \cap (X + A)$.

Para provar a *uniformidade*, note que se vale que $K' \subseteq K$, $K' \cup A$ é inconsistente sse $K' \cup B$ é inconsistente, então pelo lema B.11, $K \downarrow A = K \downarrow B$, e logo, $\bigcap \gamma(K \downarrow A) = \bigcap \gamma(K \downarrow B)$.

(postulados ⇒ construção)

Seja $*$ um operador que satisfaz os postulados AGM e seja $\gamma(K \downarrow A) = \{X \in K \downarrow A | K \cap (K * A) \subseteq X\}$ se A é construção e $\gamma(K \downarrow A) = \{K\}$ caso contrário. Vamos mostrar que: 1) $\gamma(K \downarrow A)$ é uma função de seleção e 2)$K *_\gamma A = K * A$.

1. Pelo lema B.11 e pela *uniformidade* temos que γ é bem definido, i.e., se $K \downarrow B = K \downarrow B$ então $\gamma(K \downarrow A) = \gamma(K \downarrow B)$.

   Para provar que γ é uma função de seleção temos que mostrar que se $K \downarrow A \neq \emptyset$, então $\emptyset \neq \gamma(K \downarrow A) \subseteq K \downarrow A$. Se A é inconsistente, por definição $\gamma(K \downarrow A) = \{K\}$. Caso contrário segue da *consistência* que $K * A$ é consistente. Nesse caso, $(K * A) + A$ é consistente por *fecho* e *sucesso*. Pelo lema B.10 e como $(K \cap K * A) \cup A$, existe X tal que $K \cap (K * A) \subseteq X \in K \downarrow A$. Logo $X \in \gamma(K \downarrow A)$.

2. Se A é inconsistente, segue do *fecho*, do *sucesso*, e da explosão inconsistente que, tanto $K * A$ quanto $K *_\gamma A$, são o único conjunto de crenças inconsistente.

   Se A é consistente, $K \cap (K * A) \subseteq X$ para todo $X \in \gamma(K \downarrow A)$. Segue que$K \cap (K * A) \subseteq \bigcap \gamma(K \downarrow A)$. Pela monotonicidade, $(K \cap (K * A)) + A \subseteq \bigcap \gamma(K \downarrow A) + A$ e por *fecho*, *sucesso* e *inclusão* e distributividade $K * A \subseteq K *_\gamma A$.

   Para provar que $K *_\gamma A \subseteq K * A$, mostraremos que $\bigcap \gamma(K \downarrow A) \subseteq K * A$. Seja $\beta \in \bigcap \gamma(K \downarrow A) \setminus K * A$. Como $\bigcap \gamma(K \downarrow A) \subseteq K$, pela *relevância* existe K' tal que $K \cap (K * A) \subseteq K' \subseteq K$ e $K' \cup A$ é consistente, mas $K' \cup A \cup \{\beta\}$ é inconsistente. Como $K' \subseteq K$ e $K' \cup A$ é consistente,

pelo lema B.10 existe X tal que $\beta \notin X$, $K' \subseteq X \in K \downarrow A$. Como $K \cap (K * A) \subseteq X$, pela construção de γ, $X \in \gamma(K \downarrow A)$ e, logo, $\beta \notin \bigcap \gamma(K \downarrow A)$.

□

Lema B.13. *Considere que $\langle \mathscr{L}, Cn \rangle$ é booleana. Se $K*A$ satisfaz a vacuidade e a extensionalidade, então $K * A$ também satisfaz a uniformidade.*

Demonstração. Considere que para todo $K' \subseteq K$, $K' + A$ é inconsistente sse $K' + B$ é inconsistente.

Se $K + A$ é consistente temos que $K * A = K + A$ e $K * B = K + B$ pela *vacuidade*. Nesse caso temos t $K = K \cap K * A = K \cap K * B$.

Se $K + A$ é inconsistente, pelo lema B.8 temos que $\neg A \subseteq K$ e como $\neg A + A$ é inconsistente, $\neg A + B$ também é. Segue que (de novo pelo lema B.8) $\neg B \subseteq Cn(\neg A)$. Usando argumento análogo $\neg A \subseteq Cn(\neg \beta)$. Logo, $Cn(\neg A) = Cn(\neg B)$. Isso significa que B é uma negação para A ($\neg A$). Como a lógica é distributiva temos que $Cn(A) = Cn(B)$. Da *extensionalidade* segue que $K * A = K * B$ e então $K \cap K * A = K \cap K * B$. □

Lema B.14. *Se $\langle \mathscr{L}, Cn \rangle$ satisfaz as suposições AGM então na presença dos demais postulados AGM para revisão a relevância é equivalente à identidade de Harper.*

Demonstração. **(identidade de Harper ⇒ relevância)** Se a lógica satisfaz as suposições AGM então pelo teorema de representação 6.8 temos que a relevância para contração segue dos demais postulados AGM para contração. Assim, pela identidade de Harper temos que $K - \neg A = K * A \cap K$ satisfaz a relevância para contração. Segue que se $\beta \notin K \setminus K * A$ então existe K' tal que $\neg A \nsubseteq Cn(K')$, mas $\neg A \subseteq Cn(K' \cup \{\beta\})$. Pelo lema B.8 temos que $Cn(K' \cup A)$ é consistente e $Cn(K' \cup A \cup \{\beta\})$ é inconsistente.

(relevância ⇒ identidade de Harper) Como estamos considerando que a lógica subjacente satisfaz as suposições AGM, pelo teorema de representação 6.8 temos que a recuperação e a relevância para a contração são equivalentes na presença dos demais postulados AGM para contração. Pela construção a contração $K - \neg A = K \cap K * A$ satisfaz *fecho* e *inclusão*. Como a negação é única, modulo equivalências lógicas, pela extensionalidade da revisão, a *extensionalidade* da contração também é satisfeita. Pela consistência e pelo lema B.8 $\neg A \nsubseteq K * A$. Segue que $K - \neg A$ satisfaz *sucesso*. Se $\neg A \nsubseteq K$, pelo lema B.8 $K + A$ é inconsistente. Segue da vacuidade da revisão que $K * A \cap K = K + A \cap K = K$. Logo, a consistência também satisfaz *vacuidade*. Para provar que $K - \neg A = K \cap K * A$ satisfaz a relevância, note que se $\beta \in K \setminus K - \neg A$ então $\beta \in K \setminus K * A$ e pela relevância da revisão temos K' tal que $K' \cup A$ é consistente e $K' \cup A \cup \{\beta\}$ não é. Isso significa que, pelo lema B.8, $\neg A \nsubseteq Cn(K')$, mas $\neg A \subseteq Cn(K' \cup \{\beta\})$. Portanto, $K - \neg A = K * A \cap K$ satisfaz *relevância*. □

Lema B.15. *[Mak87] Se a lógica subjacente satisfaz as suposições AGM então a identidade e Harper segue dos demais postulados AGM.*

Proposição 7.7 Se a lógica subjacente satisfaz as suposições AGM os postulados AGM para revisão são equivalentes ao seguinte conjunto de postulados: sucesso, fecho, inclusão, uniformidade e relevância.

Demonstração. Segue direto da proposição 7.5 e dos lemas B.13, B.14 e B.15. □

B.4 Provas do Capítulo 8

B.4.1 Relações entre postulados

Proposição 8.1 Na presença do sucesso forte e da inclusão a relevância apresentada na seção 8.1 é equivalente à relevância apresentada na seção 7.

Demonstração. **(relevância da seção 8.1) ⇒ (relevância da seção 7)** Se $\beta \in B \setminus B * \alpha$ então existe B' tal que $B * \alpha \subseteq B' \subseteq B \cup \{\alpha\}$ tal que B' é consistente, mas $B' \cup \{\beta\}$ não é. Considere $B'' = B' \cap B$. Temos que $B * \alpha \cap B \subseteq B'' \subseteq B$ e pelo sucesso forte $B'' \cup \{\alpha\}$ é consistente. Como $B' \subseteq B \cup \{\alpha\}$ temos que $B' \subseteq (B \cup \{\alpha\}) \cap (B' \cup \{\alpha\})$ e, portanto, $B' \subseteq B'' \cup \{\alpha\}$. Pela monotonicidade concluímos que $B'' \cup \{\alpha, \beta\}$ é inconsistente.

(relevância da seção 7) ⇒ (relevância da seção 8.1) Se $\beta \in B \setminus B * \alpha$ então existe B' tal que $B * \alpha \cap B \subseteq B' \subseteq B$ tal que B' é consistente, mas $B' \cup \{\alpha, \beta\}$ não é. Considere $B'' = B' \cup \{\alpha\}$.

Pela inclusão e pelo sucesso temos que $B * \alpha \subseteq B'' \subseteq B \cup \{\alpha\}$. Pela definição de B'' temos que $B'' \cup \{\alpha\}$ é consistente, mas $B'' \cup \{\alpha, \beta\}$ não é. □

Lema B.16 (Expansão inconsistente)**.** *[Han99] Se $B * A$ satisfaz relevância e sucesso forte então também satisfaz* expansão inconsistente*.*

(expansão inconsistente) Se A é inconsistente então $B * A = B \cup A$.

B.4.2 Revisão externa sem negação

Teorema de representação 8.6 (REKS): Uma *revisão externa kernel sem negação e com sucesso forte* (REKS) $B * A$ satisfaz sucesso forte, consistência fraca, inclusão, core-retainment e pre-expansão sse existe uma função de incisão σ que protege a entrada tal que $B * A = B + A \setminus (A, B + A \perp\!\!\!\perp \Omega)$.

Demonstração. **(construção ⇒ postulados)**

Seja σ uma função de incisão que protege a entrada e seja $B *_\sigma A = B \cup A \setminus \sigma(A, B \cup A \perp\!\!\!\perp \Omega)$. Segue direto da construção que $*_\sigma$ satisfaz *inclusão* e *pré-expansão*. O *sucesso forte* e a *consistência fraca* seguem da definição de σ. Por fim, para provar que o *core-retainment* é satisfeito, considere $\beta \in B \setminus B *_\sigma A$. Pela construção $\beta \in \sigma(A, (B \cup A) \perp\!\!\!\perp \Omega)$. Isso significa que para algum $X \in (B \cup A) \perp\!\!\!\perp \Omega$, $\beta \in X$. Seja $B' = X \setminus \{\beta\}$. Temos que $B' \subseteq B \cup A$, B' é consistente e $B' \cup \{\beta\}$ não é consistente.

(postulados ⇒ construção)

Seja $*$ uma operação que satisfaz os postulados do enunciados nesse teorema e seja σ tal que para qualquer conjunto de sentenças A:

$$\sigma(A, B \perp\!\!\!\perp \Omega) = B \setminus (B * A)$$

Temos que mostrar que (1) σ é uma função de incisão que protege a entrada e que (2) $B * A = B *_\sigma A$.

1. temos que mostrar que as três condição da definição 8.4 são satisfeitas. Para primeira condição, seja $\beta \in \sigma(A, B \perp\!\!\!\perp \Omega)$. Então temos que $\beta \in B \setminus (B * A)$ e segue do *core-retainment* que existe $B' \subseteq B + A$ tal que B' é consistente, mas $B' \cup \{\beta\}$ não é. Então existe algum subconjunto B'' de B' tal que $B'' \cup \{\beta\} \in B \perp\!\!\!\perp \Omega$ e portanto, $\beta \in \bigcup(B \perp\!\!\!\perp \Omega)$.

   Para provar a segunda condição, considere um conjunto de sentenças consistente A e pegue um X tal que $\emptyset \neq X \in B \perp\!\!\!\perp \Omega$. Suponha por contradição que $X \cap \sigma(A, B \perp\!\!\!\perp \Omega) = \emptyset$, então $X \subseteq B * A$. Como X é consistente, segue da monotonicidade que $B * A$ é consistente. Porém, isso contraria a *consistência fraca*. Essa contradição prova que $X \cap \sigma(A, B \perp\!\!\!\perp \Omega) \neq \emptyset$.

   Pelo *sucesso forte*, $A \subseteq B * A$, e logo, $A \cap \sigma(A, B \perp\!\!\!\perp \Omega) = \emptyset$.

2. Por definição, $\sigma(A, (B \cup A) \perp\!\!\!\perp \Omega) = (B \cup A) \setminus \sigma((B \cup A) * A) = (B \cup A) \setminus B * A$ (*pre-expansão*). Logo, $B *_\sigma A = (B \cup A) \setminus \sigma((B \cup A) \perp\!\!\!\perp \Omega) = (B \cup A) \setminus ((B \cup A) \setminus B * A) = B * A$ (*inclusão*).

□

Teorema de representação 8.8 (REKC): Uma *revisão externa kernel sem negação com consistência forte* (REKC) $B * A$ satisfaz sucesso fraco, consistência forte, inclusão, core-retainment e pre-expansão sse existe uma função de incisão σ que protege entradas consistentes tal que $B * A = B + A \setminus \sigma(A, B + A \perp\!\!\!\perp \Omega)$.

Demonstração. **(construção ⇒ postulados)**

Seja σ uma função de incisão que protege entradas consistentes e seja $B *_\sigma A = B \cup A \setminus \sigma(A, B \perp\!\!\!\perp \Omega)$.

A *inclusão* e a *pré-expansão* seguem direto da construção. O *sucesso fraco* e a *consistência forte* seguem da definição da função de incisão que protege entradas consistentes. Finalmente, para o *core-retainment*, seja $\beta \in B \setminus B *_\sigma A$. Pela construção, $\beta \in \sigma(A, (B \cup A) \perp\!\!\!\perp \Omega)$. Isso significa que para algum conjunto $X \in (B \cup A) \perp\!\!\!\perp \Omega$, $\beta \in X$. Seja $B' = X \setminus \{\beta\}$. Temos que $B' \subseteq B \cup A$ e como X é minimal então B' é consistente, mas $B' \cup \{\beta\}$ não é.

(postulados ⇒ construção)

Seja $*$ um operador que satisfaz os postulados do enunciado desse teorema e seja σ tal que para todo conjunto de sentenças A:

$$\sigma(A, B \perp\!\!\!\perp \Omega) = B \setminus (B * A)$$

Temos que provar que (1) σ é uma função de incisão que protege entradas consistentes e que (2) $B * A = B *_\sigma A$.

1. Devemos mostrar que as três condições da definição 8.7 são satisfeitas. Para primeira condição, seja $\beta \in \sigma(A, B \perp\!\!\!\perp \Omega)$. Então temos que $\beta \in B \setminus (B * A)$ e segue de *core-retainment* que existe algum $B' \subseteq B \cup A$ tal que B' é consistente, mas $B' \cup \{\beta\}$ não é. Então existe um subconjunto B'' de B' tal que $B'' \cup \{\beta\} \in B \perp\!\!\!\perp \Omega$. Logo, $\beta \in \bigcup(B \perp\!\!\!\perp \Omega)$.

   Para a segunda condição, seja $\emptyset \neq X \in B \perp\!\!\!\perp \Omega$. Suponha por absurdo que $X \cap \sigma(A, B \perp\!\!\!\perp \Omega) = \emptyset$. Então $X \subseteq B * A$. Como X é inconsistente, pela monotonicidade $B * A$ é inconsistente. Porém, isso contraria a *consistência forte*. Dessa contradição temos que $X \cap \sigma(A, B \perp\!\!\!\perp \Omega) \neq \emptyset$.

   Suponha que A é consistente. Pelo *sucesso fraco*, $A \subseteq B * A$ e, portanto, $A \cap \sigma(A, B \perp\!\!\!\perp \Omega) = \emptyset$.

2. Pela definição, $\sigma(A, (B \cup A) \perp\!\!\!\perp \Omega) = (B \cup A) \setminus ((B \cup A) * A) = (B \cup A) \setminus B * A$ (*pré-expansão*). Logo, $B *_\sigma A = (B \cup A) \setminus \sigma(A, (B \cup A) \perp\!\!\!\perp \Omega) = (B \cup A) \setminus ((B \cup A) \setminus B * A) = B * A$ (*inclusão*).

□

Teorema de representação 8.11 (REPMS): Uma *revisão externa partial meet sem negação e com sucesso forte* (REPMS) $B * A$ satisfaz sucesso forte, consistência fraca, inclusão, relevância e pre-expansão sse existe uma função de seleção γ que protege a entrada tal que $B * A = \bigcap \gamma(A, B \cup A \perp \Omega)$.

Demonstração. **(construção ⇒ postulados)**

Inclusão, *pré-expansão* e *consistência fraca* seguem direto da construção. Para provar *sucesso forte* e *relevância* vamos analisar dois casos:

Se A é consistente então $B * A = \bigcap \gamma(A, B \cup A \perp \Omega)$. Pelo lema B.5 existe $X \in B \cup A \perp \Omega$ tal que $A \subseteq X$. Portanto o *sucesso* é satisfeito. Considere $\beta \notin \bigcap \gamma(A, B \cup A \perp \Omega)$. Então existe $B' \in \gamma(A, B \cup A \perp \Omega)$ com $\beta \notin B'$. Pela definição B' é consistente, $B' \cup \{\beta\}$ não é consistente e $\bigcap \gamma(A, B \cup A \perp \Omega) \subseteq B' \subseteq B \cup A$.

Se A é inconsistente então pelo lema B.16 $B * A = B \cup A$ e, logo, o sucesso forte é satisfeito. A relevância é satisfeita trivialmente pois não existe $\beta \in B \setminus B \cup A$.

(postulados ⇒ construção)

Seja $*$ um operador que satisfaz os postulados do enunciado do teorema:

$$\begin{aligned} \gamma(A, B \cup A \perp \Omega) &= \{X \in B \cup A \perp \Omega : B * A \subseteq X\} \text{ se A é consistente.} \\ \gamma(A, B \cup A \perp \Omega) &= \{B \cup A\} \text{ caso contrário.} \end{aligned}$$

Temos que provar que 1) γ é uma função de seleção que protege a entrada e que 2) $B *_\gamma A = B * A$

1. Se A é consistente temos que $\gamma(A, B \cup A \perp \Omega) \neq \emptyset$. Note que pela *consistência*, pela *inclusão* e pela *propriedade do limite superior* existe $B * A \subseteq B' \in (B \cup A) \perp \Omega$.

2. Para provar que $B*A = \bigcap \gamma(A, B \cup A \perp \Omega)$ note que $\emptyset \neq B*A \subseteq X$ para todo $X \in (B \cup A) \perp \Omega$ e que $B * A \subseteq \bigcap \gamma(A, B \cup A \perp \Omega)$

   Se A é consistente então $\bigcap \gamma(A, B \cup A \perp \Omega) \subseteq B*A$ segue da *relevância* e $A \in \bigcap \gamma(A, B \cup A \perp \Omega)$ segue do *sucesso forte*.

   Se A é inconsistente então como $*$ satisfaz *sucesso*, *inclusão* e *relevância*, pelo lema B.16 $*$ também satisfaz *expansão inconsistente*. Logo, $B * A = B + A$

□

Teorema de representação 8.13 (REPMC): Uma *revisão externa partial meet sem negação e com consistência forte* (REPMC) $B*A$ satisfaz sucesso fraco, consistência forte, inclusão, relevância e pre-expansão sse existe uma função de seleção γ que protege entradas consistentes tal que $B*A = \bigcap \gamma(A, B \cup A \perp \Omega)$.

Demonstração. **(construção ⇒ postulados)**

Inclusão, *pré-expansão* e *consistência forte* seguem direto da construção. A *relevância* e o *sucesso fraco* valem pelo mesmo motivo que no teorema de representação 8.11.

(postulados ⇒ construção)

Seja $*$ que satisfaz os postulados do enunciado do teorema:

$$\gamma(A, K \cup A \perp \Omega) = \{X \in K \cup A \perp \Omega : K * A \subseteq X\}$$

Temos que mostrar que 1) γ é uma função de seleção que protege entradas consistentes e que 2) $K *_\gamma A = K * A$

1. Pela *consistência*, pela *inclusão* e pelo lema B.5 temos que $B * A \subseteq B' \in (B \cup A) \perp \Omega$. Logo, $\gamma(A, B \cup A \perp \Omega) \neq \emptyset$.

   Segue direto do *sucesso fraco* que se A é consistente então $A \in \bigcap \gamma(A, B \cup A \perp \Omega)$

2. Para mostrar que $B*A = \bigcap \gamma(A, B \cup A \perp \Omega)$ note que $\emptyset \neq B*A \subseteq X$ para todo $X \in B \cup A \perp \Omega$ e, então, $B * A \subseteq \bigcap \gamma(A, B \cup A \perp \Omega)$

   $\bigcap \gamma(A, B \cup A \perp \Omega) \subseteq B * A$ segue da *relevância* e $A \in \bigcap \gamma(A, B \cup A \perp \Omega)$ se A é consistente segue do *sucesso fraco*.

□

B.4.3 Revisão interna sem negação

Lema B.17. *Se $*$ satisfaz inclusão (para bases) então $B * A \setminus A = B \cap B * A$*

Demonstração. Como $\beta \in B * A \setminus A$ então $\beta \in B * A$, mas $\beta \notin A$. Pela inclusão temos que $\beta \in B \cup A$. Portanto $\beta \in B$ e concluímos que $\beta \in B * A \cap B$. □

Teorema de representação 8.15 (RIPM): Seja a lógica subjacente tarskiana e compacta, a revisão $*$ satisfaz inclusão, sucesso forte, consistência fraca, relevância e uniformidade sse existe alguma função de seleção γ tal que $B * A = \bigcap \gamma(B \downarrow A) \cup A$

Demonstração. **(construção $\Rightarrow$ postulados)**

Sucesso forte, *consistência fraca* e *inclusão* seguem direto da construção. A uniformidade segue do lema B.11.

Seja $B *_\gamma A = \bigcap \gamma(B \downarrow A) + A$ e seja $\beta \in B \setminus B *_\gamma A$. Então existe $X \in \gamma(B \downarrow A)$ tal que $\beta \notin X$. Pela definição de $B \downarrow A$ temos que $B \cap B *_\gamma A \subseteq X \subseteq B$, $X \cup A$ é consistente, mas $X \cup A \cup \{\beta\}$ é inconsistente.

(postulados $\Rightarrow$ construção) Seja $*$ uma revisão que satisfaz os postulados enunciados no teorema:

$$\begin{aligned} \gamma(B \downarrow A) &= \{X \in B \downarrow A : B \cap B * A \subseteq X\} \text{ se } A \text{ é consistente} \\ \gamma(B \downarrow A) &= \{B\} \text{ caso contrário} \end{aligned}$$

Vamos provar que 1) γ é uma função de seleção e 2) que $B *_\gamma A = B * A$.

1. Se $B \downarrow A = B \downarrow A'$ pelo lema B.11 e pela *uniformidade* $\gamma(B \downarrow A) = \gamma(B \downarrow A')$.

   Se A é consistente pela *consistência fraca* $B * A$ é consistente. Então, por *inclusão* e *sucesso forte* $(B \cap B * A) \cup A$ é consistente e pelo lema B.10 temos que existe $X \in B \downarrow A$ tal que $B \cap B * A \subseteq X$. Logo, $\gamma(B \downarrow A) \neq \emptyset$.

2. $B \cap B * A \subseteq X$ para todo $X \in \gamma(B \downarrow A)$. Segue que $B \cap B * A \subseteq \bigcap \gamma(B \downarrow A)$. Pelo lema B.17 $B * A \setminus A \subseteq \bigcap \gamma(B \downarrow A)$. Portanto $B * A \subseteq B *_\gamma A$

   Seja $\beta \in \bigcap \gamma(B \downarrow A) \setminus B * A$. Como $\bigcap \gamma(B \downarrow A) \subseteq B$, pela *relevância* existe B' tal que $B \cap B * A \subseteq B' \subseteq B$, $B' \cup A$ é consistente, mas $B' \cup A \cup \{\beta\}$ é inconsistente. Pelo lema B.10 existe $X \in B \downarrow A$ tal que $B' \subseteq X$ e $\beta \notin X$. Segue que $B \cap B * A \subseteq B \subseteq X \in B \downarrow A$. Logo, $X \in \gamma(B \downarrow A)$. Como $\beta \notin X$, $\beta \notin \gamma(B \downarrow A)$. Porém, isso contradiz a definição de β. Logo não existe $\beta \in \bigcap \gamma(B \downarrow A) \setminus B * A$. Ou seja, $B *_\gamma A \subseteq B * A$.

□

Lema B.18. *Seja $\langle \mathscr{L}, Cn \rangle$ uma lógica compacta e monotônica. Então $B{\downdownarrows}A = B{\downdownarrows}A'$ sse para todo $B' \subseteq B$, $B' \cup A$ é inconsistente sse $B' \cup A'$ é inconsistente.*

Demonstração. ($\Rightarrow$) Suponha por absurdo que exista $B' \subseteq B$ tal que $B' \cup A$ é inconsistente, mas $B' \cup A'$ é consistente. Pela compacidade e pela monotonicidade temos que existe $B'' \subseteq B'$ tal que $B'' \in B{\downdownarrows}A$. Como $B' \cup A'$ é consistente, $B'' \cup A'$ também é e, portanto, $B'' \notin B{\downdownarrows}A'$ contradizendo a hipótese.

($\Leftarrow$) Suponha por absurdo que $X \in B{\downdownarrows}A$, mas $X \notin B{\downdownarrows}A'$. Temos então dois casos:

1. $X \cup A'$ é consistente. Temos então que $X \cup A$ é inconsistente, mas $X \cup A'$ não é, contradizendo a hipótese.

2. $X \cup A'$ é inconsistente. Como $X \notin B{\downdownarrows}A'$ deve existir $X' \subset X$ tal que $X' \cup A'$ é inconsistente. Nesse caso $X' \subset X \in B{\downdownarrows}A$, portanto $X' \cup A$ é consistente contrariando a hipótese.

□

Teorema de representação 8.19 (RIK): Seja a lógica subjacente tarskiana e compacta, a operação de revisão $*$ satisfaz sucesso forte, consistência fraca, inclusão, uniformidade e core-retainment sse $B * A = (B \setminus \sigma(B\!\downdownarrows\!A)) \cup A$ para alguma função de incisão σ.

Demonstração. **(construção $\Rightarrow$ postulados)**

Sucesso forte, *consistência fraca* e *inclusão* seguem direto da construção.

Seja $\beta \in B \setminus B * A$. Então $\beta \in \cup(B\!\downdownarrows\!A)$ e, portanto, existe X tal que $\beta \in X \in \sigma(B\!\downdownarrows\!A)$. Segue que $(X \setminus \{\beta\}) \cup A$ é consistente, mas $X \cup A$ é inconsistente. Portanto a *relevância* é satisfeita.

A *uniformidade* segue direto do lema B.18.

(postulados $\Rightarrow$ construção) Seja $*$ uma revisão que satisfaz os postulados enunciados no teorema:

$$\sigma(B\!\downdownarrows\!A) = B \setminus B * A$$

Seja $B *_\sigma A = (B \setminus \sigma(B\!\downdownarrows\!A)) \cup A$. Vamos provar que 1) σ é uma função de incisão e 2) que $B *_\sigma A = B * A$.

1. Se $B\!\downdownarrows\!A = B\!\downdownarrows\!A'$ então pelo lema B.18 e pela *uniformidade* temos que $B * A = B * A'$ e portanto a função é bem definida.

   Se $\beta \in \sigma(B\!\downdownarrows\!A)$ então $\beta \in B \setminus B * A$. Por *core-retainment* existe $B' \subseteq B$ tal que $B' \cup A$ é consistente, mas $B' \cup A \cup \{\beta\}$ não é. Por compacidade e monotonicidade existe $B'' \subseteq B'$ tal que $B'' \cup \{\beta\} \in B\!\downdownarrows\!A$. Logo $\beta \in \bigcup B\!\downdownarrows\!A$.

   Seja $\emptyset \neq X \in B\!\downdownarrows\!A$ e suponha por absurdo que $X \cap \sigma(B\!\downdownarrows\!A) = \emptyset$. Então $X \subseteq B * A$ e como $X \cup A$ é inconsistente, segue da monotonicidade que $B * A$ é inconsistente. Como $\emptyset \neq X \in B\!\downdownarrows\!A$, A é consistente. Pela *consistência fraca* temos um absurdo. Logo, $X \cap \sigma(B\!\downdownarrows\!A) \neq \emptyset$.

2. $B *_\sigma A = B \setminus \sigma(B\!\downdownarrows\!A) \cup A = B \setminus (B \setminus B * A) \cup A = (B \cap B * A) \cup A$. Segue que $B *_\sigma A \subseteq B * A \cup A$. Pelo *sucesso* $B *_\sigma A \subseteq B * A$. De $B *_\sigma A = (B \cap B * A) \cup A$ pela *inclusão* e pelo lema B.17 temos que $B *_\sigma A = (B * A \setminus A) \cup A$. Pelo *sucesso* $B * A \subseteq B *_\sigma A$.

□

B.5 Provas do Capítulo 9

Lema B.19. *Seja $\mathcal{B}$ uma classe de conjuntos e $X \subseteq B$ para todo $X \in \mathcal{B}$ então todo corte mínimo C de $\mathcal{B}$ satisfaz $C \subseteq B$.*

Demonstração. Suponha por absurdo que C é um corte de $\mathcal{B}$ e existe $\beta \in C$ tal que $\beta \notin B$. Como $\beta \notin B$ e $X \subseteq B$ para todo $X \in \mathcal{B}$ temos que β não pertence a nenhum elemento de $\mathcal{B}$. Portanto $C \setminus \{\beta\} \cap X' \neq \emptyset$ para todo $X' \in \mathcal{B}$. Como $S \setminus \{\beta\} \subset S$ temos que S não é um corte minimal contrariando a hipótese. □

Lema B.20. *Se $X \subseteq B$ e $Y = B \setminus X$ então $X = B \setminus Y$.*

Demonstração. Simples manipulação de conjuntos. □

Teorema 9.2 Seja $\mathscr{L}$ uma lógica tarskiana e compacta, $B \subseteq \mathscr{L}$ e $\alpha \in \mathscr{L}$:

1. $X \in \{B \setminus X : X \in B \perp\!\!\!\perp \alpha\}$ sse X é um corte mínimo de $B \perp\!\!\!\perp \alpha$
2. $X \in B \perp\!\!\!\perp \alpha$ sse X é um corte mínimo de $\{B \setminus X : X \in B \perp\!\!\!\perp \alpha\}$

Demonstração. 1. **(X é um corte mínimo de $B \perp\!\!\!\perp \alpha \Rightarrow X \in \{B \setminus X : X \in B \perp\!\!\!\perp \alpha\}$)**

Como X é um corte mínimo de $B \perp\!\!\!\perp \alpha$ pelo lema B.19 temos que $X \subseteq B$. Portanto existe $Y \subseteq B$ tal que $X = B \setminus Y$. Temos que mostrar que $Y \in B \perp\!\!\!\perp \alpha$.

- $Y \subseteq B$ é trivial
- $Y = B \setminus X$ pelo lema B.20. Portanto pelo sucesso da contração kernel temos que $\alpha \notin Cn(Y)$.
- Se $X' \subset X$ então $\alpha \in Cn(B \setminus X')$ pois, como X é um corte mínimo de $B \perp\!\!\!\perp \alpha$, existe $K \in B \perp\!\!\!\perp \alpha$ tal que $X' \cap K = \emptyset$. Logo $K \subseteq B \setminus X'$ uma vez que $K \subseteq B$. Concluímos que $\alpha \in Cn(Y \cup \{\beta\})$ para qualquer $\beta \in B \setminus X$.

($X \in \{B \setminus Y : Y \in B \perp \alpha\} \Rightarrow X$ é um corte mínimo de $B \perp\!\!\!\perp \alpha$)

Seja $Y \in B \perp \alpha$ e $X = B \setminus Y$. Pelo lema B.20 temos que $Y = B \setminus X$. Tenho que mostrar que X é um corte mínimo para $B \perp\!\!\!\perp \alpha$

- Suponha por absurdo que existe $K \in B \perp\!\!\!\perp \alpha$ tal que $K \cap X = \emptyset$. Então $K \subseteq B \setminus X$ e, portanto, $\alpha \in Cn(B \setminus X) = Cn(Y)$ que viola a definição de Y. Concluímos que para todo $K \in B \perp\!\!\!\perp \alpha$ temos que $K \cap X \neq \emptyset$.
- Seja $X' \subset X$, então $\alpha \in Cn(B \setminus X')$ então X' não é corte de $B \perp\!\!\!\perp \alpha$ pelo sucesso da contração kernel. Concluímos que X é um corte mínimo de $B \perp\!\!\!\perp \alpha$.

2. **(X é um corte mínimo de $\{B \setminus X : X \in B \perp \alpha\} \Rightarrow X \in B \perp\!\!\!\perp \alpha$)**

   Temos que mostrar que se X é um corte mínimo de $\{B \setminus X : X \in B \perp \alpha\}$ então $X \in B \perp\!\!\!\perp \alpha$

   - Pelo lema B.19 temos que $X \subseteq B$
   - Suponha por absurdo que $X \subseteq Y \in B \perp \alpha$. Então $(B \setminus Y) \cap X = \emptyset$ contrariando a definição de Y. Logo não existe $X \subseteq Y \in B \perp \alpha$. Suponha agora que $\alpha \notin Cn(X)$. Como $X \subseteq B$ pela propriedade upper-bound temos que existe $Y \in B \perp \alpha$ tal que $X \subseteq Y$, mas mostramos acima que isso é impossível. Logo, $\alpha \in Cn(X)$.
   - Suponha por absurdo que existe $X' \subset X$ tal que $\alpha \in Cn(X')$. Nesse caso X' seria um corte de $\{B \setminus Y : Y \in B \perp \alpha\}$ o que seria um absurdo pois X é um corte mínimo.

   ($X \in B \perp\!\!\!\perp \alpha \Rightarrow X$ é um corte mínimo de $\{B \setminus X : X \in B \perp \alpha\}$) Seja $X \in B \perp\!\!\!\perp \alpha$. Temos que mostrar que X é um corte mínimo de $\{B \setminus X : X \in B \perp \alpha\}$.

   - Suponha por absurdo que existe $B \setminus Y$ tal que $Y \in B \perp \alpha$ e $(B \setminus Y) \cap X = \emptyset$. Como $X \subseteq B$ e $Y \subseteq B$ teríamos $Y \subseteq X$ de $(B \setminus Y) \cap X = \emptyset$, mas isso contraria o fato de que $\alpha \in Cn(X)$, mas $\alpha \notin Cn(Y)$.
   - Suponha por absurdo que existe $X' \subset X$ tal que $(B \setminus Y) \cap X' \neq \emptyset$ para todo $Y \in B \perp \alpha$. Nesse caso teríamos que $\alpha \in Cn(X')$, pois caso contrário pela propriedade upper-bound existiria Y' tal que $X' \subseteq Y' \in B \perp \alpha$ e nesse caso $(B \setminus Y) \cap X' = \emptyset$ o que contraria a definição.

□

Referências Bibliográficas

[ACKZ09] A. Artale, D. Calvanese, R. Kontchakov, e M. Zakharyaschev. The DL-lite family and relations. Relatório Técnico BBKCS-09-03, School of Computer Science and Information Systems, Birbeck College, 2009. Disponível em http://www.dcs.bbk.ac.uk/research/techreps/2009/bbkcs-09-03.pdf. 48

[AGM85] C. Alchourrón, P. Gärdenfors, e D. Makinson. On the logic of theory change. *Journal of Symbolic Logic*, 50(2):510–530, 1985. i, iii, 4, 5, 7, 17, 19, 20, 23, 24, 25, 26, 27, 56, 59, 72, 74, 76, 88, 115

[AM81] C. Alchourrón e D. Makinson. Hierarchies of regulation and their logic. Em Hilpinen, editor, *New studies in deontic logic*, páginas 125–148. D. Reidel Publishing Company, 1981. 63, 113

[AM82] C. Alchourron e D. Makinson. On the logic of theory change: Contraction functions and their associated revision functions. *Theoria*, 48:14–37, 1982. 27, 31

[Are00] C. Areces. *Logic Engineering. The Case of Description and Hybrid Logics*. Tese de Doutorado, Institute for Logic, Language and Computation, University of Amsterdam, Amsterdam, Netherlands, Outubro 2000. 50

[BBL05] F. Baader, S. Brandt, e C. Lutz. Pushing the EL envelope. Em L. P. Kaelbling e A. Saffiotti, editors, *International Joint Conferences on Artificial Intelligence (IJCAI 05)*, páginas 364–369, Edinburgh, Scotland, UK, 2005. Professional Book Center. 48

[BCM+03] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, e P. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003. 35, 42

[BLHL01] T. Berners-Lee, J. Hendler, e O. Lassila. The Semantic Web: A new form of web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American*, Maio, 2001. 6

[BS01] F. Baader e U. Sattler. An overview of tableau algorithms for description logics. *Studia Logica*, 69(1):5–40, October 2001. 39, 41, 42

[BVL03] S. Bechhofer, R. Volz, e P. W. Lord. Cooking the semantic web with the OWL API. Em D. Fensel, K. P. Sycara, e J. Mylopoulos, editors, *Proceedings of the 2nd International Semantic Web Conference (ISWC 2003)*, páginas 659–675. Springer, October 20-23 2003. 95

[CGL+05] D. Calvanese, G. D. Giacomo, D. Lembo, M. Lenzerini, e R. Rosati. DL-lite: Tractable description logics for ontologies. Em M. M. Veloso e S. Kambhampati, editors, *Proceedings of the 20th National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference (AAAI 2005)*, páginas 602–607, Pittsburg, Pennsylvania, USA, Julho 9-13 2005. AAAI Press / The MIT Press. 48

[Coo71] S. A. Cook. The complexity of theorem-proving procedures. Em *Proceedings of the 3rd annual ACM Symposium on Theory of Computing (STOC '71)*, páginas 151–158, Shaker Heights, Ohio, USA, 1971. ACM. 14, 60

[Del08] J. P. Delgrande. Horn clause belief change: Contraction functions. Em G. Brewka e J. Lang, editors, *Proceedings of the 10th International Conference on the Principles of Knowledge Representation and Reasoning (KR'08)*, páginas 156–165, Sydney, Australia, Setembro 16-19 2008. AAAI Press. 77, 114, 115

[Den87] D. C. Dennett. *The Intentional Stance*. MIT Press, 1987. 3

[DLNN97] F. M. Donini, M. Lenzerini, D. Nardi, e W. Nutt. The complexity of concept languages. *Information and Computation*, 134:1–58, 1997. 39

[DSB+04] M. Dean, G. Schreiber, S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, e L. A. Stein. OWL web onotology language reference. W3C Recomendation, 2004. 47

[dSW07] T. de Sousa e R. Wassermann. Handling inconsistencies in CTL model-checking using belief revision. Em *Proceedings of the Tenth Brazilian Symposium in Formal Methods*, Ouro Preto, Minas Gerais, Brasil, 2007. 63

[DW10] J. Delgrande e R. Wassermann. Horn clause contraction functions: Belief set and belief base approaches. Em F. Lin, U. Sattler, e M. Truszczynski, editors, *Proceedings of the 12th International Conference on the Principles of Knowledge Representation and Reasoning (KR'10)*, Toronto, Canada, Maio, 9-13 2010. AAAI Press. 61

[FBdC85] M. Fattorosi-Barnaba e F. de Caro. Graded modalities I. *Studia Logica*, 44(2):197–221, Junho 1985. 51

[FFKI06] M. A. Falappa, E. L. Fermé, e G. Kern-Isberner. On the logic of theory change: Relations between incision and selection functions. Em G. Brewka, S. Coradeschi, A. Perini, e P. Traverso, editors, *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI 2006)*, páginas 402–406, Riva del Garda, Italy, 2006. IOS Press. 94

[FH94] A. Fuhrmann e S. O. Hansson. A survey of multiple contractions. *Journal of Logic, Language and Information*, 3(1):39–75, Janeiro 1994. 24

[FH99] E. L. Fermé e S. O. Hansson. Selective revision. *Studia Logica*, 63(3):331–342, 1999. 106

[FHP+06] G. Flouris, Z. Huang, J. Z. Pan, D. Plexousakis, e H. Wache. Inconsistencies, negations and changes in ontologies. Em *Proceedings of the 21st National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference (AAAI 2006)*, Boston, Massachusetts, USA, July 16-20 2006. AAAI Press. 43

[Flo06] G. Flouris. *On Belief Change and Ontology Evolution*. Tese de Doutorado, University of Crete, 2006. xi, 5, 7, 24, 26, 53, 54, 55, 57, 63, 64, 66, 74, 75, 76, 103

[FPA05] G. Flouris, D. Plexousakis, e G. Antoniou. On applying the AGM theory to DLs and OWL. Em E. M. Yolanda Gil, V. R. Benjamins, e M. A. Musen, editors, *Proceedings of the 4th International Semantic Web Conference (ISWC 2005)*, páginas 216–231, Galway, Ireland, November, 6-10 2005. Springer. 5, 53, 63, 74, 75, 76

[FPA06] G. Flouris, D. Plexousakis, e G. Antoniou. On generalizing the AGM postulates. Em *Proceedings of the 3rd European Starting AI Researcher Symposium (STAIRS-06)*, 2006. 5, 24, 69, 70, 71, 72, 74, 75, 76

[Fuh91] A. Fuhrmann. Theory contraction through base contraction. *Journal of Philosophical Logic*, 20(2):175–203, Maio 1991. 28

[Fuh97] A. Fuhrmann. *An Essay on Contraction*. CSLI Publications, Stanford University, USA, 1997. 5, 24

[Gra71] G. Gratzer. *Lattice Theory: First Concepts And Distributive Lattices*. W. H. Freeman and co, San Francisco, California, USA, 1971. 56

[Gro88] A. Grove. Two modellings for theory change. *Journal of Philosophical Logic*, 17:157–170, 1988. 4, 24

[Gru93] T. R. Gruber. Towards Principles for the Design of Ontologies Used for Knowledge Sharing. Em N. Guarino e R. Poli, editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Deventer, Netherlands, 1993. Kluwer Academic Publishers. 6

[GW10] P. T. Guerra e R. Wassermann. Revision of CTL models. Em *Advances in Artificial Intelligence - Proceedings of IBERAMIA 2010*, volume 6433 of *Lecture Notes in Computer Science*, páginas 153–162. Springer, 2010. 63

[Gär88] P. Gärdenfors. *Knowledge in Flux: Modeling the Dynamics of Epistemic States*. MIT Press, Cambridge, UK, 1988. 4, 7, 17, 18, 19, 24, 26, 27

[Han89] S. O. Hansson. New operators for theory change. *Theoria*, 55:115–132, 1989. 106

[Han91] S. O. Hansson. Belief contraction without recovery. *Studia Logica*, 50(2):251–260, 1991. 22, 23, 29, 69, 70, 71

[Han93] S. O. Hansson. Reversing the Levi identity. *Journal of Philosophical Logic*, 22:637–669, 1993. 31, 78, 81, 88, 132

[Han94] S. O. Hansson. Kernel contraction. *Journal Symbolic Logic*, 59(3):845–859, 1994. 30

[Han97] S. O. Hansson. Semi-revision (invited paper). *Journal of Applied Non-Classical Logics*, 7(2), 1997. 33, 82, 90

[Han99] S. O. Hansson. *A Textbook of Belief Dynamics*. Kluwer Academic, 1999. 5, 7, 17, 27, 28, 29, 105, 109, 117

[Har86] G. Harman. *Change In View: Principles of Reasoning*. Bradford Books. MIT Press, 1986. 6, 18

[HKS06] I. Horrocks, O. Kutz, e U. Sattler. The even more irresistible SROIQ. Em *Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR'06)*, páginas 57–67, Lake District, UK, Junho 2006. AAAI Press. 47

[HM01] V. Haarslev e R. Möller. Racer system description. Em *Proceedings of the International Joint Conference on Automated Reasoning (IJCAR 2001)*, volume 2083 of *Lecture Notes in Artificial Intelligence*, páginas 701–705. Springer, 2001. 47

[HNSS90] B. Hollunder, W. Nutt, e M. Schmidt-Schauß. Subsumption algorithms for concept description languages. Em A. L., editor, *Proceedings of the 9th European Conference on Artificial Intelligence (ECAI 1990)*, páginas 348–353, Stockholm, Sweden, 1990. Pitman. 39

[Hor98] I. Horrocks. Using an expressive description logic: FaCT or fiction? Em A. G. Cohn, L. Schubert, e S. C. Shapiro, editors, *Proceedings of the 6th International Conference on the Principles of Knowledge Representation and Reasoning (KR'98)*, páginas 636–647, Trento, Italia, Junho, 2-5 1998. Morgan Kaufmann. 47

[HP06] J. Hladik e R. Peñaloza. PSPACE automata for description logics. Em B. Parsia, U. Sattler, e D. Toman, editors, *Proceedings of the 2006 Workshop on Description Logics (DL2006)*, volume 189 of *CEUR Workshop Proceedings*, Windermere, Lake District, UK, Maio 30-Junho 1 2006. CEUR-WS.org. Disponível em http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-189/. 41

[HS04] P. Haase e Y. Sure. State-of-the-art on ontology evolution. SEKT informal deliverable 3.1.1.b, Institute AIFB, University of Karlsruhe, 2004. 103

[HST00] I. Horrocks, U. Sattler, e S. Tobies. Practical reasoning for very expressive description logics. *Logic Journal of the IGPL*, 8, 2000. 45

[HW02] S. O. Hansson e R. Wassermann. Local change. *Studia Logica*, 70(1):49–76, 2002. 8, 30, 31, 32, 57

[HWKP06] C. Halaschek-Wiener, Y. Katz, e B. Parsia. Belief base revision for expressive description logics. Em *Proceedings of the OWL: Experiences and Directions (OED'06)*, 2006. 90

[Kal06] A. Kalyanpur. *Debugging and repair of owl ontologies.* Tese de Doutorado, University of Maryland, College Park, MD, USA, 2006. 90, 91, 92, 93

[KM92] F. Katsuno e A. Mendelzon. On the difference between updating a knowledge database and revising it. Em P. Gardenfors, editor, *Belief Revision.* Cambridge University Press, 1992. 18

[KPSH05] A. Kalyanpur, B. Parsia, E. Sirin, e J. Hendler. Debugging unsatisfiable classes in OWL ontologies. *Journal of Web Semantics*, 3(4), 2005. 91

[KPTR+06] F. Kepler, C. Paz-Trillo, J. Riani, M. M. Ribeiro, K. Valdivia-Delgado, L. N. de Barros, e R. Wassermann. Classifying ontologies. Em *Proceedings of the 2nd Workshop on Ontologies and their Applications (WONTO 2006)*, 2006. 6, 104

[LB87] H. J. Levesque e R. J. Brachman. Expressiveness and tractability in knowledge representation and reasoning. *Computational Intelligence*, 3:78–93, 1987. 4, 35

[Lev91] I. Levi. *The Fixation of Belief and its Undoing: Changing Beliefs through Inquiry.* Cambridge University Press, Outubro 1991. 23, 69

[LSST08] M. Langlois, R. H. Sloan, B. Szörényi, e G. Turán. Horn complements: Towards Horn-to-Horn belief revision. Em D. Fox e C. P. Gomes, editors, *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI 2008)*, páginas 466–471, Chicago, Illinois, USA, Julho 13-17 2008. AAAI Press. 61

[Lut02] C. Lutz. *The Complexity of Description Logics with Concrete Domains.* Tese de Doutorado, Fakultät für Mathematik, Informatik und Naturwissenschaften der Rheinisch, Hamburg, Germany, Março 2002. 45

[Mak87] D. Makinson. On the status of the postulate of recovery in the logic of theory change. *Journal of Philosophical Logic*, 16:383–394, 1987. 22, 23, 26, 27, 69, 116

[MGH+08] B. Motik, B. C. Grau, I. Horrocks, Z. Wu, A. Fokoue, e C. Lutz. OWL 2 web ontology language: Profiles. World Wide Web Consortium, Working Draft, Dezembro 2008. Disponível em http://www.w3.org/TR/2008/WD-owl2-profiles-20081202/. 48

[Min81] M. Minsky. A framework for representing knowledge. Em J. Haugeland, editor, *Mind Design: Philosophy, Psychology, Artificial Intelligence*, páginas 95–128. MIT Press, Cambridge, MA, 1981. 5, 35, 48

[MS88] J. P. Martins e S. C. Shapiro. A model for belief revision. *Artificial Intelligence*, 35(1):25–79, 1988. 28

[MSH07] B. Motik, R. Shearer, e I. Horrocks. Optimized Reasoning in Description Logics using Hypertableaux. Em F. Pfenning, editor, *Proceedings of the 21st Conference on Automated Deduction (CADE-21)*, volume 4603 of *LNAI*, páginas 67–83, Bremen, Germany, July 17-20 2007. Springer. 47, 95

[Neb89] B. Nebel. A knowledge level analysis of belief revision. Em R. J. Brachman, H. J. Levesque, e R. Reiter, editors, *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning (KR'89)*, páginas 301–311, Toronto, Canada, Maio, 15-18 1989. Morgan Kaufmann. 106

[Neb90a] B. Nebel. *Reasoning and revision in hybrid representation systems*. Springer-Verlag New York, Inc., New York, NY, USA, 1990. 4, 28

[Neb90b] B. Nebel. Terminological reasoning is inherently intractable. *Artificial Intelligence*, 43(2):235–249, 1990. 41

[Nie91] R. Niederée. Multiple contraction. a further case against gärdenfors' principle of recovery. Em *Proceedings of the Workshop on The Logic of Theory Change*, páginas 322–334, London, UK, 1991. Springer-Verlag. 23, 69

[OWL09] W. OWL Working Group. *OWL 2 Web Ontology Language: Document Overview*. W3C Recommendation, 27 October 2009. Disponível em `http://www.w3.org/TR/owl2-overview/`. 47

[Pea88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988. 17

[PS10a] R. Peñaloza e B. Sertkaya. Complexity of axiom pinpointing in the dl-lite family of description logics. Em H. Coelho, R. Studer, e M. Wooldridge, editors, *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010)*, volume 215 of *Frontiers in Artificial Intelligence and Applications*, páginas 29–34, Lisbon, Portugal, 2010. IOS Press. 97

[PS10b] R. Peñaloza e B. Sertkaya. On the complexity of axiom pinpointing in the el family of description logics. Em F. Lin, U. Sattler, e M. Truszczynski, editors, *Proceedings of the 12th International Conference on the Principles of Knowledge Representation and Reasoning (KR'10)*, Toronto, Canada, Maio, 9-13 2010. AAAI Press. 97

[PSK05] B. Parsia, E. Sirin, e A. Kalyanpur. Debugging OWL ontologies. Em *Proceedings of the 14th international conference on World Wide Web (WWW 2005)*, páginas 633–640, New York, NY, USA, 2005. ACM Press. 91

[QHHP08] G. Qi, P. Haase, Z. Huang, e J. Z. Pan. A kernel revision operator for terminologies. Em F. Baader, C. Lutz, e B. Motik, editors, *Proceedings of the 21st Workshop on Description Logics (DL2008)*, volume 353 of *CEUR Workshop Proceedings*, Dresden, Germany, Maio 13-16 2008. CEUR-WS.org. Disponível em http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-353/. 90

[Qui67] R. M. Quillian. Word concepts: A theory and simulation of some basic semantic capabilities. *Behavioral Science*, 12:410–430, 1967. 5, 35, 48

[Rei87] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, 1987. 91, 93

[RN03] S. Russell e P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2 edição, 2003. 3

[Rot92] H. Rott. On the logic of theory change: More maps between different kinds of contraction functions. Em P. Gardenfors, editor, *Belief Revision*. Cambridge University Press, 1992. 4, 24

[RRP96] A. L. Rector, J. E. Rogers, e P. A. Pole. The GALEN high level ontology. Em *Proceedings of Medical Informatics Europe 96*, páginas 174–178. IOS Press, Janeiro 1996. 48

[RW06] M. M. Ribeiro e R. Wassermann. First steps towards revising ontologies. Em *Proceedings of the 2nd Workshop on Ontologies and their Applications (WONTO 2006)*, 2006. 6, 103

[RW07] M. M. Ribeiro e R. Wassermann. Base revision in description logics - preliminary results. Em *Proceedings of the International Workshop on Ontology Dynamics (IWOD 2007)*, Innsbruck, Austria, Junho, 7 2007. 7, 104

[RW08a] M. M. Ribeiro e R. Wassermann. Degrees of recovery and inclusion in belief base dynamics. Em M. Pagnucco e M. Thielscher, editors, *Proceedings of the 12th International Workshop on Non-Monotonic Reasoning (NMR 2008), Sidney, Australaia, Setembro 13-15, 2008*, Sidney, Australia, Setembro 13-15 2008. 7

[RW08b] M. M. Ribeiro e R. Wassermann. The ontology reviser plug-in for protégé. Em *Proceedings of the 3rd Workshop on Ontologies and their Applications (WONTO 2008), Salvador, Brazil, October 26, 2008*, 2008. 7, 104

[RW08c] M. M. Ribeiro e R. Wassermann. On the relation between remainder sets and kernels. Em *XV Encontro Brasileiro de Lógica (EBL08)*, Paraty, RJ, Brasil, 2008. Resumo espandido. 7, 104

[RW09a] M. M. Ribeiro e R. Wassermann. AGM revision in description logics. Em *Proceedings of the 1st Workshop on Automated Reasoning about Context and Ontology Evolution (ARCOE 2009)*, Pasadena, California, USA, Julho, 11-12 2009. 7, 103

[RW09b] M. M. Ribeiro e R. Wassermann. Base revision for ontology debugging. *Journal of Logic and Computation*, 19:721–743, 2009. Special Issue: Recent Advances in Ontology Dynamics. 7, 104

[RW10] M. M. Ribeiro e R. Wassermann. More about agm revision in description logics. Em *Proceedings of the 2nd Workshop Automated Reasoning about Context and Ontology Evolution (ARCOE 10)*, Lisboa, Portugal, Agosto, 16-17 2010. 7, 103

[RWA+09] M. M. Ribeiro, R. Wassermann, G. Antoniou, G. Flouris, e J. Pan. Contraction on the semantic web: On the role of relevance and recovery. Em *Proceedings of the International Workshop on Ontology Dynamics (IWOD 2009)*, Washington DC, USA, Outubro, 26 2009. 7, 103

[SC03] S. Schlobach e R. Cornet. Non-standard reasoning services for the debugging of description logic terminologies. Em G. Gottlob e T. Walsh, editors, *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI 03)*, páginas 355–362, Acapulco, Mexico, 2003. Morgan Kaufmann. 90, 91, 92

[Sch91] K. Schild. A correspondence theory for terminological logics: preliminary report. Em J. Mylopoulos e R. Reiter, editors, *Proceedings of the 12th international joint conference on Artificial intelligence (IJCAI 91)*, páginas 466–471, Sydney, New South Wales, Australia, 1991. Morgan Kaufmann. 41, 50

[Smi82] B. C. Smith. *Reflection and Semantics in a Procedural Language.* Tese de Doutorado, Massachusetts Institute of Technology, 1982. 4

[Smu68] R. M. Smullyan. *First Order Logic.* Springer-Verlag, 1968. 13, 14, 15

[SPG+07] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, e Y. Katz. Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics*, 5(2):51–53, 2007. 47, 95

[SR04] K. A. Spackman e G. Reynoso. Examining snomed from the perspective of formal ontological principles: Some preliminary analysis and observations . Em U. Hahn, editor, *1st International Workshop on Formal Biomedical Knowledge Reprsentation (KR-MED)*, volume 102 of *CEUR Workshop Proceedings*, páginas 72–80, Whistler, BC, Canada, Junho, 1 2004. CEUR-WS.org. Disponível em http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-102/. 48

[SSS91] M. Schmidt-Schaubße G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991. 41

[Sta84] R. C. Stalnaker. *Inquiry.* The MIT Press, Cambridge, MA, 1984. 6

[Sto04] L. Stojanovic. *Methods and Tools for Ontology Evolution.* Tese de Doutorado, University of Karlsruhe, Germany, 2004. 103

[TH06] D. Tsarkov e I. Horrocks. Fact++ description logic reasoner: System description. Em *Proceedings of the International Joint Conference on Automated Reasoning (IJCAR 2006)*, volume 4130 of *Lecture Notes in Artificial Intelligence*, páginas 292–297. Springer, 2006. 47, 95

[Tob00] S. Tobies. The complexity of reasoning with cardinality restrictions and nominals in expressive description logics. *Journal of Artificial Intelligence Research*, 12, 2000. 46

[Tob01] S. Tobies. Complexity results and practical algorithms for logics in knowledge representation. *Computing Research Repository (CoRR)*, cs.LO/0106031, 2001. Disponível em http://arxiv.org/abs/cs/0106031v1. 41

[Was00a] R. Wassermann. An algorithm for belief revision. Em A. G. Cohn, F. Giunchiglia, e B. Selman, editors, *Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR'00)*, páginas 345–352, Breckenridge, Colorado, USA, Abril, 15-20 2000. Morgan Kaufmann. 91, 93

[Was00b] R. Wassermann. *Resource Bounded Belief Revision.* Tese de Doutorado, Universiteit van Amsterdan, Janeiro 2000. 5, 7, 28, 33, 93

Índice Remissivo

$*$, 19
$+$, 19
$-$, 19
$\cdot^{\mathcal{I}}$, 37
?, 33
$A^{-}(K)$, 54
B^{*}, 106
C_{Horn}, 60
C_{LPC}, 12
C_{LTL}, 62
C_{gl}, 49
C_{loc}, 49
Cn, 5
Cn^{*}, 106
G, 62
K, 5
K_n, 48
N_C, 36
N_I, 36
N_R, 36
N_{R_+}, 44
X, 62
Δ, 107
$\Delta^{\mathcal{I}}$, 37
$\Delta^{\mathcal{I}}_{\mathbf{D}}$, 45
Δ_f, 108
$\mathcal{I}$, 37
$\mathscr{L}_{Horn}$, 60
$\mathscr{L}_{K_n}$, 49
$\mathscr{L}_{LPC}$, 11
Ω, 81
$\downarrow$, 77
$\equiv$, 36
γ, 25, 86, 87
γ^{*}, 106
$\Downarrow$, 89
$\perp\!\!\!\perp$, 30
$\mathbb{P}$, 11, 49
$\mathbb{R}$, 49
$\mathcal{H}_N$, 48
$\mathscr{L}$, 5
$\perp$, 24, 73
σ, 30, 84, 85
$\sqsubseteq$, 36
$\dot{\sqsubseteq}$, 44
$\models^{loc}$, 49
$\models_{\mathcal{M}}$, 61
$\models_{v}$, 11
$\models_{\mathcal{M}}$, 49
$\models_{gl}$, 49
sup, 54
v, 11
$\mathcal{ALC}$, 36
$\mathcal{ALCO}$, 50
$\mathcal{H}$, 44
$\mathcal{O}$, 45
$\mathcal{Q}$, 45
$\mathcal{SROIQ}$, 47
$\mathcal{S}$, 44
ínfimo, 54
$\mathcal{SHIF}(\mathbf{D})$, 47
$\mathcal{SHOIN}(\mathbf{D})$, 47
$\langle P, \leq \rangle$, 54
$\langle \mathscr{L}, Cn \rangle$, 5
(D), 45

adequação epistêmica, 4
adequação heurística, 4
agente, 3
agente racional, 3
algoritmo black-box, 91
algoritmo glass-box, 91
assinatura, 11
atitudes epistêmicas, 17
atualização de crenças, 18

base de conhecimento, 4, 37
base de conhecimento satisfatível, 43

caráter finito, 14
compacidade, 14
compatibilidade, 74
conceito satisfatível, 42
conjunto de crenças, 19
conjunto kernel, 30

conjunto resíduo, 24, 73
conjuntos complementares, 54
consistência, 26, 31
consistência forte, 82
consistência fraca, 82
consolidação, 32
construtores, 36
contração AGM, 19
contração kernel, 30
contração partial meet, 25
contração-relevante, 70
core-addition, 107
core-retainment, 29, 31, 82
corte, 93
corte máximo, 55
corte mínimo, 93
crença, 3
critério da minimalidade, 18

decomponibilidade, 54
dedução, 15
distributividade, 56

entrada, 20
entrada epistêmica, 18
equivalência lógica, 11
estado epistêmico, 3, 17
expansão, 4
expansão AGM, 19
expansão inconsistente, 117
explosão inconsistente, 20
extensionalidade, 21, 26, 28

fórmulas bem formadas, 11, 53
fecho, 21, 26
fecho por negação (clássica), 56
finitamente representável, 24
forma normal negada, 13
função de incisão, 30
    que protege a entrada, 84
    que protege entradas consistentes, 85
função de seleção, 25
    protege entradas consistentes, 87
    que protege a entrada, 86

hierarquia de papéis, 44
hipótese da representação do conhecimento, 4

identidade de Harper, 26
inclusão, 22, 26, 28, 31, 83
inclusão lógica, 106
interpretação canônica, 14

lógica
    idempotente, 12
    inclusão, 12
    monotônica, 12
    tarskiana, 12
lógica AGM compatível, 54
lógica booleana, 56
lógica de descrição, 35
    inconsistência, 43
lógica fechada por negação clássica, 56
lógica subjacente, 19
lógicas de descrição
    internalização, 46
leis de De Morgan, 13
lema de Lindembaum, 15
lema de Tuckey, 14
linguagem, 11, 53
linguagem de descrição, 36

método de decisão, 14
modelo, 11
mudanças de crenças, 18

não contravenção local α, 32
não contravenção local A, 57
negação clássica, 56
nominais, 45
NP-completo, 60

operações, 18
operador de consequência, 11, 53
ordem parcial, 54
OWL, 46

papéis inversos, 44
papéis transitivos, 44
papel concreto, 45
postulados AGM para contração, 23
postulados de racionalidade, 18
pré-expansão, 32, 83
procedimento de prova, 12
propagação de descrença, 28
propriedade da cadeia descendente (ascendente), 57
propriedades lógicas, 53
Protégé, 95
pseudo-contração partial meet geral, 106
pseudo-contrações, 106

recuperação, 23
recuperação ótima, 70
recuperação lógica, 106
relevância, 29, 31, 70, 78, 82
relevância lógica, 107
representação de conhecimento, 4

restrição numérica qualificada, 45
reticulado completo, 54
revisão, 4
revisão AGM, 19
revisão externa, 31
revisão externa kernel
    sem negação, 83
        com consistência forte, 86, 118
        com sucesso forte, 85, 117
revisão externa kernel:, 32
revisão externa partial meet
    sem negação, 86
        com consistência forte, 87, 88, 119
        com sucesso forte, 87, 118
revisão externa partial meet:, 32
revisão interna, 31
revisão interna kernel
    sem negação, 89
revisão interna kernel:, 32
revisão interna partial meet
    sem negação, 88
revisão interna partial meet:, 32
revisão partial meet, 27

satisfatibilidade lógica, 11
semi-revisão, 32
sentenças, 11
sistema de crenças, 18
sistema intencional, 3
sub-reticulado, 56
sucesso, 21, 26, 28, 31
sucesso forte, 82
sucesso fraco, 82
suposições AGM, 20
supra-classicalidade:, 20
supremo, 54

tableau
    completo, 13
    conflitos, 13
    fechado, 13
    ramo, 13
taxonomia, 41
teorema de representação, 18, 25
teoria das bases de crenças, 27
troca interna, 33

uniformidade, 29, 31, 88
uniformidade [Han93], 78
uniformidade fraca, 32

vacuidade, 22, 26, 28
validade lógica, 11
valorações, 11
variáveis proposicionais, 11

web-semântica, 6
withdrawal, 69