IVAN JOSÉ VARZINCZAK

CAUSALIDADE E DEPENDÊNCIA EM RACIOCÍNIO SOBRE AÇÕES

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Informática, Curso de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.

Orientador: Marcos Alexandre Castilho

CURITIBA

2002

IVAN JOSÉ VARZINCZAK

CAUSALIDADE E DEPENDÊNCIA EM RACIOCÍNIO SOBRE AÇÕES

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Informática, Curso de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.

Orientador: Marcos Alexandre Castilho

CURITIBA

2002

TERMO DE APROVAÇÃO

IVAN JOSÉ VARZINCZAK

CAUSALIDADE E DEPENDÊNCIA EM RACIOCÍNIO SOBRE AÇÕES

Dissertação aprovada como requisito parcial à obtenção do grau de Mestre no Programa de Pós-Graduação em Informática da Universidade Federal do Paraná pela seguinte banca examinadora:

Orientador: Marcos Alexandre Castilho
Departamento de Informática, UFPR

Prof. Dr. Marcelo Finger
Instituto de Matemática e Estatística, USP

Prof. Dr. Michel Gagnon
Departamento de Informática, UFPR

Curitiba, 06 de junho de 2002

À professora Analice (*in memorian*), com quem aprendi os primeiros passos em lógica.

Agradecimentos

Primeiramente, quero agradecer a minha família, que, mesmo sem saber o que é minha pesquisa e quando e para quem ela pode ser útil e porque eu troquei um excelente salário numa empresa para seguir carreira acadêmica, soube compreender minha motivação e meus sonhos e sempre me deu o apoio necessário para que eu pudesse fazer aquilo de que realmente gosto.

Agradeço ao Marcos pelo inestimável apoio e atenção recebidos ao longo de todo esse tempo. Seja como aluno de graduação, como bolsista do PET ou mestrando, sempre tive a certeza de que estava muito bem orientado.

Independente das correrias do dia-a-dia, graças à sua maneira de orientar conseguimos realizar um trabalho legal. Certamente, ele é uma das poucas pessoas com quem uma rápida conversa de corredor pode valer mais do que horas de reunião.

Agradeço a ele todos os comentários, críticas e principalmente o bom exemplo que tive e as portas que me foram abertas para o futuro.

Quero agradecer também a Andreas Herzig por suas valorosas sugestões e espero que possamos realizar um bom trabalho em meu doutorado.

Agradeço o apoio recebido da CAPES, na forma de bolsa de mestrado, que me permitiu poder me dedicar inteiramente à pesquisa.

Um agradecimento especial aos responsáveis pela grande greve de 2001 (!), sem a qual eu certamente não teria tido o tempo necessário para escrever a maior parte deste trabalho.

Realmente é impossível listar os nomes de todas as pessoas que são importantes para mim e que direta ou indiretamente contribuíram comigo para que eu também pudesse dar minha parcela de contribuição. Se você é uma delas (e certamente vai saber se é), então lhe agradeço agora!

Finalmente, não poderia deixar de agradecer também a Deus, mas isso eu continuarei fazendo pelo resto de minha vida...

Ivan José Varzinczak

Curitiba, junho de 2002

Inútil, na batalha, vencer milhões de inimigos. Vencer-se a si mesmo é a maior de todas as vitórias.

— Budha

Sumário

Lista de Figuras

Resumo

Neste trabalho, propomos uma solução para os problemas da persistência e da ramificação em termos de uma relação de dependência que indica em que circunstâncias uma determinada ação pode alterar o valor de verdade de um dado literal. Esta relação pode ser vista como uma noção causal fraca, noção esta considerada fundamental para raciocinar sobre ações corretamente. Uma vez integrada ao formalismo modal da Lógica de Ações e Planos $\mathcal{LAP}$, obtém-se uma abordagem causal simples e poderosa para tratar problemas que envolvem ações. Neste texto, nós fazemos uma comparação sistemática entre outros formalismos recentes que tratam do mesmo problema. Mostramos também como integrar a dependência ao formalismo modal de $\mathcal{LAP}$, estabelecendo provas de adequação e completude do sistema axiomático e do método de prova baseado em tableaux semânticos aqui definidos. Finalmente, através de exemplos clássicos da área e de outros novos aqui propostos, mostramos as vantagens de nossa abordagem em relação a outras encontradas na literatura.

Palavras-chave: Representação do conhecimento, raciocínio sobre ações, lógica modal, causalidade, relação de dependência, sistemas de tableau.

Abstract

In this work we propose a solution to the frame and ramification problems in terms of a dependence relation that points out the circumstances in which a certain action may change the truth value of a given literal. Such relation can be viewed as a weak causal notion, which is considered fundamental for correctly reasoning about actions. Once integrated in the base formalism of the Logic of Actions and Plans $\mathcal{LAP}$, it gives us a simple and powerful causal approach to deal with problems involving actions. In this text we make a systematic comparison among other recent formalisms with the same purposes. We also show how to integrate the dependence relation in the modal formalism of $\mathcal{LAP}$, establishing soundness and completeness proofs for the axiomatic system and the tableau based proof method so defined. Finally, by means of some classical scenarios of the area and other ones that we propose here, we show the advantages of our approach with respect to other formalisms in the literature.

Keywords: Knowledge representation, reasoning about actions, modal logics, causality, dependence relation, tableau systems.

Résumé

Dans ce travail, nous proposons une solution pour les problèmes du décor e de la ramification en termes d'une relation de dépendance qui spécifie les circonstances dans lesquelles une certaine action peut changer la valeur de vérité d'un littéral donné. Cette relation peut être vue comme une notion de causalité faible, ce qui est considéré fondamental pour raisonner correctement sur les actions. Une fois intégrée au formalisme modal de la Logique des Actions et des Plans $\mathcal{LAP}$, nous obtenons une approche causale simple et puissante pour traiter des problèmes en concernant des actions. Dans ce texte, nous faisons une comparaison systématique avec d'autres formalismes récents qui traitent de cette problématique. Nous montrons aussi comment intégrer la dépendance au formalisme modal de $\mathcal{LAP}$ et présentons les preuves d'adéquation et de complétude du système axiomatique et de la méthode d'inférence basée sur les tableaux sémantiques définis ici. Finalement, par le biais de quelques exemples classiques du domaine et d'autres que proposons ici, nous montrons les avantages de notre approche par rapport aux autres trouvées dans la littérature.

Mots-clés: Représentation de la connaissance, raisonnement sur les actions, logique modale, causalité, relation de dépendance, système de tableau.

Parte I

Apresentação

Capítulo 1

Introdução

Toda ação é precedida por um pensamento.

— Ralph Waldo Emerson

Desde o surgimento da Inteligência Artificial nos anos 50, as pesquisas nessa área vêm sendo conduzidas no sentido de formalizar as sutilezas do raciocínio humano, de maneira a poder reproduzi-lo automaticamente. Sendo assim, um de seus objetivos fundamentais é possibilitar a construção de máquinas dotadas da capacidade de realizar tarefas consideradas como inteligentes. A fim de atingir esse objetivo, uma das principais ferramentas de base utilizada para a representação do conhecimento e a realização de inferências é a *lógica matemática*, conforme sugerido por McCarthy [43].

Os principais argumentos pró e contra o uso da lógica em Inteligência Artificial foram compilados por Shanahan [63], que mostrou que toda e qualquer forma de representação pode ser reduzida à lógica formal. Isso reforça a hipótese de que a lógica é uma boa maneira de formalizar o conhecimento e o processo de raciocínio humanos.

Em certos domínios, no entanto, o uso da lógica formal leva a problemas difíceis de serem tratados e que por vezes estão em aberto há longos anos. Um desses casos, conhecido como *raciocínio sobre ações*, é objeto de estudo do presente trabalho.

1.1 O que é raciocinar sobre ações

Uma propriedade fundamental de uma máquina destinada a raciocinar como um ser humano e a agir de forma inteligente é a capacidade de raciocinar sobre um mundo que pode mudar em função das ações que são executadas. Isso envolve a representação de fatos acerca de um mundo dinâmico e a realização de inferências nessa representação. Tais inferências consistem em considerar o estado do mundo antes da execução de uma certa ação, os efeitos produzidos pela mesma, bem como o estado resultante de sua execução.

Neste contexto, para ser considerado inteligente, um robô precisaria ser capaz de empreender um processo de raciocínio a respeito das ações que podem ser realizadas e das conseqüências de suas execuções, sejam elas diretas ou indiretas, levando em conta as condições necessárias para sua realização.

Como exemplo, um robô deve saber que se há um grande vazamento de gás na cozinha, riscar um fósforo pode provocar um incêndio e matar o bebê que está no segundo andar. Logo, ele será considerado inteligente se, ao invés de riscar o fósforo, cortar o vazamento e abrir as portas e janelas.

A definição de inteligência, nesse caso, está intimamente relacionada ao comportamento, ou seja, à realização de tarefas consideradas como inteligentes.

1.2 Tarefas em raciocínio sobre ações

Três tipos de tarefas que um agente inteligente deve ser capaz de realizar constituem interesse na área de raciocínio sobre ações: a *predição*, a *explicação* e a *planificação*.

A predição (figura 1.1) consiste em "prever" o estado do mundo *após* a execução de uma determinada seqüência de ações. Como exemplo, suponhamos que estamos numa sala escura e que podemos executar a ação *apertar o interruptor*. A tarefa de predição, nesse caso, é inferir que após a ação de apertar o interruptor, a sala estará iluminada.

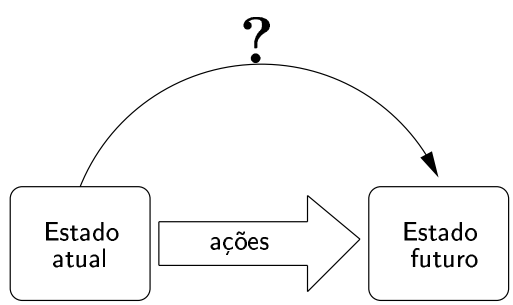

Figura 1.1: Predição — raciocinar sobre o futuro.

A explicação (figura 1.2), por seu turno, é o contrário da predição, isto é, consiste em determinar o estado do mundo *antes* da execução de uma certa seqüência de ações. Como exemplo, suponhamos que estamos numa sala iluminada e que sabemos que o interruptor foi acionado. Intuitivamente, isso nos permite concluir que antes a sala estava escura.

Já a planificação (figura 1.3) é a tarefa de saber se existe uma seqüência de ações que possa nos levar de um dado estado do mundo a outro e qual é tal seqüência. Em outros termos, a planificação consiste em determinar as ações necessárias para se atingir um determinado objetivo. Como exemplo, suponhamos que estamos numa sala escura

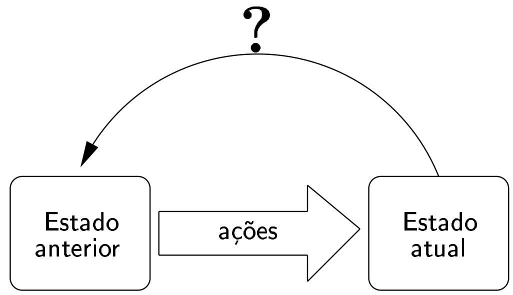

Figura 1.2: Explicação — raciocinar sobre o passado.

e que desejamos que a mesma se torne iluminada. Nesse caso, através da planificação procuramos uma seqüência de ações que nos leve de uma situação na qual a sala está escura a uma situação na qual a mesma está iluminada[1]. Para esse exemplo, as soluções possíveis seriam a ação de abrir uma janela ou a de apertar o interruptor.

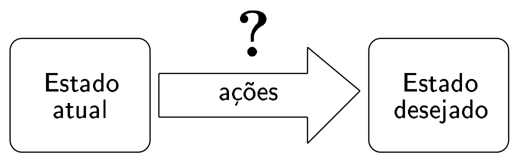

Figura 1.3: Planificação — o que fazer para atingir um objetivo.

Realizar as tarefas de predição, explicação e planificação é mais ou menos trivial dependendo do formalismo de base adotado. Neste texto, nós investigaremos a problemática envolvida quando se adota a lógica formal matemática.

1.3 Problemas que surgem em raciocínio sobre ações

Três importantes problemas são inerentes ao uso da lógica matemática em representação do conhecimento e raciocínio sobre ações e serão brevemente descritos nesta seção: *o problema da persistência*, *o problema da ramificação* e *o problema da qualificação*.

O problema da persistência[2], apontado pela primeira vez em 1969 por McCarthy e Hayes [47], consiste no fato de que, ao se representar um mundo dinâmico usando lógica formal, além de explicitar todos os fatos sobre as propriedades que mudam em função da execução das ações consideradas, também ser necessário enunciar *todos* os fatos sobre as propriedades que *não* mudam.

[1]Obviamente, pode haver mais de uma seqüência de ações que produzem o mesmo resultado.

[2]Na formulação original em inglês, tal problema é conhecido como *frame problem* e comumente traduzido em português como "problema do quadro". Entretanto, por considerarmos que o termo persistência denota melhor a essência desse problema, propomos neste trabalho a adoção da expressão "problema da persistência". Para detalhes a respeito da origem da expressão *frame problem*, ver [63, seção 1.1].

Como um exemplo bem simples, consideremos um domínio envolvendo a ação de apertar um interruptor. Um dos efeitos observáveis dessa ação é que a lâmpada se acende. Entretanto, há uma série de propriedades que não se alteram pelo simples apertar de um interruptor, como, por exemplo, a cor do mesmo, a posição da lâmpada, a forma da porta ou o material de que é feita a janela. Apertar o interruptor não muda o tempo que faz lá fora nem provoca (a princípio) uma guerra nuclear, entre outras coisas.

Por mais evidente que isso seja para nós humanos, em lógica não há como fazer inferências como essas tomando por base apenas as informações sobre as coisas que mudam. Neste sentido, faz-se necessária a inclusão de fatos explicitando as coisas que não mudam por causa de certas ações.

O grande problema é que a lista de todas essas não-mudanças pode acabar se tornando extremamente longa. Isso significa que, em domínios mais complexos, a quantidade de informação a ser codificada pode extrapolar a capacidade de memória de qualquer computador, além de tornar infactível o processo de realização de inferências. Esse é o problema da persistência.

O problema da ramificação, por sua vez, é fundamentado na necessidade de se empreender o raciocínio sobre todas as possíveis conseqüências da execução de uma determinada ação [17]. O problema, nesse caso, surge da dificuldade de se descrever *todas* essas conseqüências (ramificações) em lógica formal.

Como exemplo, considerando ainda a ação de apertar um interruptor, temos como efeitos a alteração de sua posição, o acender da lâmpada, o fato de a sala tornar-se iluminada, o de a mesma não estar mais escura, a ativação do detector de luz na parede, entre outros. A necessidade de se encontrar uma maneira econômica de representar todos os efeitos (diretos e indiretos) de uma certa ação e de poder inferi-los constitui o núcleo do problema da ramificação.

Já o problema da qualificação diz respeito à necessidade de se determinar todas as pré-condições necessárias para que uma dada ação possa ser executada com sucesso [44].

O problema, nesse caso, reside no fato de que a lista envolvendo todas as pré-condições de uma dada ação também pode ser muito extensa. Como exemplo, as qualificações da ação de acender a lâmpada abrangem o fato de o interruptor não estar bloqueado, a existência de um fio ligando o interruptor e a lâmpada, a passagem de corrente elétrica no momento da execução da ação, entre inúmeras outras. Realmente pode-se imaginar que a totalidade de tais pré-condições, apesar de finita em número, tende a ser enorme.

No próximo capítulo, retornaremos ainda aos problemas da persistência, da ramificação e da qualificação a fim de estabelecer uma definição mais precisa e detalhada em termos formais que deixará evidente a sua importância para a área.

Na seqüência, apresentamos um breve histórico das tentativas empreendidas em raciocínio sobre ações para a obtenção de soluções para seus problemas.

1.4 Evolução histórica

Historicamente, o problema da persistência foi o primeiro a ser largamente investigado pela comunidade. Como formalismo de base, a grande maioria das abordagens utilizou o *cálculo de situações*, o qual consiste de uma lógica de primeira ordem e que foi utilizado por McCarthy e Hayes na formulação original do problema [47].

Com as poucas tentativas desenvolvidas ao longo da década de 70, notadamente a de Sandewall [58], pôde-se perceber o verdadeiro grau de dificuldade que o problema da persistência apresenta.

Em 1980 surgiram dois formalismos importantes que marcaram a década: a *lógica default* [55] e a *circunscrição* [45]. Esta última, juntamente com o cálculo de situações, tornou-se um verdadeiro marco na área.

Em essência, a circunscrição consiste numa forma de raciocínio não-monotônico baseada em técnicas de minimização de modelos. Tal processo se dá mediante o enunciado de uma fórmula de segunda ordem (que será melhor descrita na seção 2.2.1), a qual, integrada a um formalismo de primeira ordem, como, por exemplo, o cálculo de situações, e partindo do princípio de que as coisas mudam minimamente, propõe que se minimize um certo predicado de maneira que a propriedade por ele representada seja alterada minimamente após a execução de uma dada ação, de acordo com uma política pré-determinada.

Inicialmente, a política original de circunscrição causou um grande impacto nas pesquisas na área no sentido em que possibilitou reduzir a uma única fórmula todo o conjunto de enunciados a respeito das não-mudanças do mundo.

Entretanto, a utilização da circunscrição apresenta o inconveniente de requerer o enunciado de fórmulas em lógica de segunda ordem. Essa particularidade, além de aumentar a complexidade de possíveis soluções ao problema da persistência, ainda sujeita a implementação de provadores de teoremas circunscritivos ao problema da parada [29], uma vez que a lógica de segunda ordem é indecidível [52].

Além disso, na medida em que as pesquisas foram evoluindo na área, surgiram vários contra-exemplos à aplicação das políticas de circunscrição propostas até então, comprovando que as mesmas não eram completas. Um dos contra-exemplos mais famosos é o cenário do tiro em Yale [27], que consiste numa situação bastante simples envolvendo seqüências de ações com uma ação sem efeito em que a aplicação da política de circunscrição proposta em [46] não produz o resultado esperado.

Além dos problemas relatados acerca da completude das diferentes políticas de circunscrição, um ponto chave que despertou grande reflexão sobre o tema foi o fato de não se conseguir implementar provadores de teoremas circunscritivos que fossem realmente eficientes para se tratar o caso geral, como é o caso das implementações propostas em [22, 54]. Mesmo que se deixasse de lado os contra-exemplos que a cada nova solução surgiam e se procurasse restringir as aplicações a casos mais simples, os algoritmos propostos eram ineficientes para certas classes de problemas [66].

Como conseqüência, no restante da década de 80 e ao longo da de 90 os pesquisadores se lançaram numa busca intensa para corrigir os métodos de circunscrição. Esta técnica foi largamente estudada e expandida em formas de diferentes graus de complexidade, as quais foram compiladas em [36].

Ao mesmo tempo, vários outros formalismos, mais ou menos complexos e poderosos, foram apresentados como substitutos ideais para o cálculo de situações. Dentre os mais importantes, destacam-se a linguagem $\mathcal{A}$ [21], o formalismo de *Features and Fluents* [59], o *cálculo de fluentes* [64], o *cálculo de eventos* [63] e a utilização de *lógicas modais* [32]. Tais formalismos foram adaptados para suportar técnicas de minimização similares às da circunscrição ou até mesmo ela própria.

Sendo assim, a pesquisa na área tornou-se uma série de propostas de soluções seguidas de contra-exemplos para os formalismos baseados nessas teorias. Numa tentativa de tornar mais simples as comparações entre os diferentes formalismos de base, passou-se a investigar em maiores detalhes as chamadas *abordagens sistemáticas*, representadas pela linguagem $\mathcal{A}$ e pelo *Features and Fluents*. A idéia fundamental por detrás de tais abordagens consistia em, ao invés de apresentar contra-exemplos, provar teoremas de completude sobre traduções de um formalismo em outro e ao mesmo tempo estabelecer os limites de cada formalismo em termos de *classes de problemas*.

Entretanto, conforme já comentado, essas novas abordagens ainda se baseavam em técnicas de minimização, estando, portanto, sujeitas aos mesmos problemas resultantes da aplicação das políticas de circunscrição existentes. Além disso, as abordagens sistemáticas não conseguiram cumprir seu papel de formalismo de base simples na representação de domínios mais complexos, em especial os que envolvem ações não-determinísticas.

Dentro desse contexto, durante a década de 90, passou a haver uma distinção cada vez mais perceptível entre os pesquisadores que procurariam ainda insistir numa abordagem não-monotônica, mediante aplicações de diferentes políticas de circunscrição a casos mais simples, e aqueles que investiriam numa alternativa monotônica para tentar solucionar os problemas da persistência e da ramificação.

As pesquisas com base em abordagens não-monotônicas, apesar dos problemas de eficiência, conseguiram obter uma solução para o problema da persistência, tendo por base o princípio de minimização dos efeitos das ações. Entretanto, tal técnica impõe uma série de dificuldades ao se tentar abordar o problema da ramificação.

As abordagens monotônicas, por seu turno, também obtiveram relativo sucesso em termos de solução para o problema da persistência, mas as propostas sugeridas para o problema da ramificação estavam muito longe de uma solução satisfatória [63].

Sendo assim, ambos os grupos se não esbarravam em detalhes de implementação que comprometiam a eficiência de cada um de seus métodos, toda vez em que obtinham uma solução para o problema da persistência, surgia um contra-exemplo que invalidava sua aplicação ao problema da ramificação.

Observou-se, então, que as dificuldades encontradas surgiam por causa da maneira com que se abordava o problema da ramificação. Até então, era consenso na comunidade que para solucionar tal problema era necessário dividir os efeitos de uma ação em efeitos diretos e indiretos, descobrir uma fórmula estática relacionando os literais e determinar uma maneira automática de deduzir os efeitos indiretos a partir dos diretos.

Com esse método, surgiam facilmente contra-exemplos apresentando situações em que ocorriam mudanças espontâneas, ou seja, que não eram causadas. Em razão disso, estabeleceu-se o consenso de que uma noção de *causalidade* se fazia necessária. Em outros termos, mudanças só poderiam ocorrer se fossem realmente causadas, e não fruto de alterações espontâneas.

Nesse sentido, o objetivo passou a ser procurar maneiras de se expressar uma noção de causa envolvendo efeitos de ações em lógica formal. Com esse princípio de causalidade por base, muitas propostas surgiram durante a década de 90, dentre as quais destacam-se o fecho de explicação [61, 62], as regras causais [41, 42], a minimização da causalidade [37] e a pós-derivação de efeitos [64, 65]. Com isso, a definição de uma noção causal precisa e de maneiras de se lidar com ela passou a constituir o núcleo das principais pesquisas empreendidas desde então.

No final da década de 90, seguindo o princípio de causalidade, foi proposta em [7] uma abordagem monotônica fundamentada em lógica modal que demonstrou ser uma das mais completas soluções para o problema da persistência apresentadas até então e que se comporta de maneira satisfatória frente ao problema da ramificação.

Em essência, a referida abordagem define um formalismo enriquecido de uma noção de *dependência* para representar a causalidade. A dependência, nesse caso, consiste de uma *noção causal fraca* envolvendo ações e seus respectivos efeitos, e é representada por uma relação binária entre ações e literais.

Com esse trabalho, obteve-se um formalismo capaz de modelar cenários não tratados pelas outras abordagens citadas até então e que, ao contrário delas, possui real aplicabilidade em situações de interesse prático.

1.5 Objetivos do trabalho

Apesar do sucesso do trabalho apresentado em [7], há situações em que a utilização da dependência binária requer o enunciado explícito de algumas informações a respeito das não-mudanças do mundo. Isso tem constituído uma crítica à referida abordagem por parte da comunidade desde sua apresentação.

Nesse sentido, o objetivo deste trabalho consiste em mostrar que definindo-se uma nova relação de dependência como uma relação *ternária* envolvendo ações, literais e fórmulas, pode-se obter uma solução melhor para os problemas em raciocínio sobre ações. Neste caso, o terceiro elemento na nova relação definida consistirá do *contexto* no qual a ação é executada e servirá para restringir os efeitos das ações nos literais considerados.

Tendo essa idéia como base, elaboraremos toda a caracterização teórica do novo formalismo resultante da definição de uma relação ternária, apresentando sua sintaxe, semântica, axiomática e complexidade computacional. Nesse sentido, mostraremos que com o novo formalismo definido, será possível obter uma representação mais flexível e fiel à realidade, em comparação às demais propostas presentes na literatura.

A fim de dotar de um caráter prático essa nova abordagem, apresentaremos ainda um método de prova adequado e completo baseado nos sistemas de tableaux semânticos.

1.6 Organização da dissertação

O presente trabalho é organizado da seguinte maneira: No capítulo 2, definimos operacionalmente os problemas da persistência, da ramificação e da qualificação no formalismo do cálculo de situações [47], que é o formalismo de base mais extensamente utilizado na literatura. Discutimos também acerca das características necessárias para que uma proposta de solução a esses problemas seja considerada satisfatória.

Nos capítulos 3–6, que constituem a segunda parte deste trabalho, apresentamos quatro abordagens que, acreditamos, resumem bem o estado da arte em raciocínio sobre ações, ilustrando os principais avanços obtidos bem como as dificuldades encontradas. Nessa parte, apresentamos duas abordagens consideradas não-monotônicas e duas monotônicas, com o intuito de esboçar as principais diferenças entre ambos os paradigmas.

Na terceira parte, apresentamos a nossa proposta para raciocinar sobre ações. Nesse sentido, definimos um formalismo de base para representação do conhecimento e apresentamos também um método de prova para a realização de inferências.

Na última parte, elaboramos uma comparação sistemática entre nossa proposta e as principais abordagens presentes na literatura, ilustrando os pontos em que o formalismo aqui desenvolvido se comporta de maneira melhor. Finalizando, tecemos algumas discussões acerca das possíveis extensões à abordagem aqui proposta, apresentando as principais conclusões que podem ser obtidas a partir deste trabalho.

Capítulo 2

Problemas em raciocínio sobre ações

— *Houston, temos um problema...*

— Apolo 13

No presente capítulo, apresentamos em maiores detalhes os principais problemas que surgem em raciocínio sobre ações ao se empreender a representação de um domínio e a realização de inferências em lógica formal.

Inicialmente, formalizamos o cálculo de situações, o qual tem sido amplamente utilizado como lógica de base pela grande maioria das abordagens propostas até então. Em seguida, utilizando o formalismo do cálculo de situações, definimos formalmente os problemas da persistência, da ramificação e da qualificação, esboçados na seção 1.3. Por fim, são estabelecidos os critérios para que uma dada solução a esses problemas possa realmente ser considerada como satisfatória.

2.1 O cálculo de situações

Originalmente, o cálculo de situações foi definido em 1969 por McCarthy e Hayes [47] e desde então diferentes variantes vêm sendo amplamente utilizadas como formalismo de base para representação de domínios em raciocínio sobre ações.

Nesta seção, apresentamos brevemente a versão do cálculo de situações que será utilizada no decorrer deste trabalho. Para um maior nível de detalhe, recomenda-se [63].

2.1.1 Definições e semântica

O cálculo de situações utiliza uma linguagem multitipada baseada na lógica de predicados de primeira ordem com igualdade [48]. Define-se um tipo SIT para as situações, um tipo FLU para os fluentes e um tipo $\mathsf{AÇÕES}$ para as ações.

Uma *situação* é uma descrição completa do estado do mundo em um certo momento. A situação inicial normalmente é representada por s_0. Um *fluente* é uma propriedade[1] que assume diferentes valores de verdade em diferentes situações. Uma *ação* consiste de um nome de uma ação em particular, cuja execução é realizada em uma dada situação e que tem a característica de potencialmente alterar os valores dos fluentes.

A linguagem do cálculo de situações possui ainda um predicado binário especial *Holds*, cujos argumentos são um fluente e uma situação. Se $f \in \mathsf{FLU}$ e $s \in \mathsf{SIT}$, então $Holds(f, s)$ representa o fato de que f é verdadeiro em s. $\neg Holds(f, s)$ significa que f é falso em s.

O cálculo de situações apresenta também a função *result*, que toma como argumentos uma ação e uma situação e tem como resultado uma outra situação. Nesse sentido, $result(\alpha, s)$ denota a situação resultante da execução da ação α na situação s.

Outro predicado definido para o cálculo de situações é *Poss*, o qual toma como argumento uma ação e uma situação e indica que a ação em questão é *possível*, isto é, *executável*, na dada situação. Um exemplo de fórmula envolvendo o predicado *Poss* é:

$$Poss(atirar, result(carregar, s_0)),$$

que denota que "é possível atirar na situação resultante de se carregar uma arma".

Por meio do predicado *Holds* e da função *result*, pode-se empreender a descrição do comportamento das ações. Isso se faz através de três tipos especiais de fórmulas: as *leis de efeito das ações*, as *leis do domínio* e as *observações*.

As leis de efeito de uma determinada ação descrevem o comportamento da mesma numa dada situação. Uma lei de efeito de ação apresenta uma das seguintes formas

$$\forall s(A \to Holds(f, result(\alpha, s)))$$
$$\forall s(A \to \neg Holds(f, result(\alpha, s))),$$

onde f é um fluente, α uma ação, s uma situação e A ou é $\top$ (em cujo caso costuma-se omitir) ou é uma fórmula clássica na qual toda e qualquer ocorrência de uma situação está numa fórmula da forma $Holds(g, s)$ ou $\neg Holds(g, s)$, onde g é um fluente.

A fórmula abaixo constitui um exemplo de lei de efeito de ação. Por questões de simplificação, convencionaremos que em todas as fórmulas complexas apresentadas, as variáveis que denotam situações estão universalmente quantificadas.

$$Holds(Carregada, s) \to \neg Holds(Vivo, result(atirar, s))$$

Tal fórmula exprime a idéia segundo a qual "se uma arma estiver carregada, então a ação de atirar mata a vítima".

[1] Uma proposição ou um termo de primeira ordem.

As leis do domínio, também chamadas de restrições de domínio, descrevem leis gerais que não dependem de ocorrência de ações, ou seja, são fórmulas que são sempre verdadeiras. Tais leis são da forma

$$\forall s A,$$

onde A é uma fórmula na qual toda e qualquer ocorrência de uma situação está numa fórmula da forma $Holds(g, s)$ ou $\neg Holds(g, s)$, onde g é um fluente. Um exemplo de uma lei de domínio é a fórmula

$$Holds(Vivo, s) \leftrightarrow \neg Holds(Morto, s),$$

que exprime o fato de que "nenhuma pessoa pode estar ao mesmo tempo viva e morta".

As observações, por seu turno, servem para descrever instâncias de situações. São fórmulas nas quais toda ocorrência de $Holds$ é da forma $Holds(f, s')$ ou $\neg Holds(f, s')$, onde s' é uma situação específica, isto é, uma variável livre (não quantificada). Exemplos de observações são as seguintes fórmulas, onde s_0 denota a situação inicial:

$$Holds(Carregada, s_0) \tag{2.1}$$

$$\neg Holds(Vivo, result(esperar, result(atirar, s_0))) \tag{2.2}$$

A fórmula (2.1) representa o fato de que "na situação inicial, a arma está carregada". A fórmula (2.2) expressa que "na situação resultante de, na situação inicial, atirar e em seguida esperar, a vítima estará morta".

A seguir, apresentamos um exemplo de formalização de um mundo dinâmico utilizando o cálculo de situações.

2.1.2 Um cenário-exemplo no cálculo de situações

Um dos exemplos mais típicos de modelagem envolvendo o cálculo de situações é o clássico mundo dos blocos [51]. Considere que tenhamos três blocos amarelos dispostos sobre uma mesa, tal qual ilustrado pela figura 2.1.

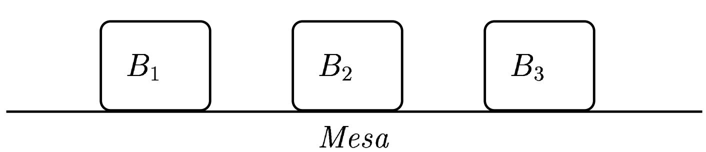

Figura 2.1: Uma configuração possível do mundo dos blocos.

Sendo assim, temos as seguintes observações, onde $Livre(x)$ significa que não há outro bloco sobre o denotado por x:

$$Holds(Sobre(B_1, Mesa), s_0) \tag{2.3}$$

$$Holds(Sobre(B_2, Mesa), s_0) \tag{2.4}$$

$$Holds(Sobre(B_3, Mesa), s_0) \tag{2.5}$$

$$Holds(Livre(B_1), s_0) \tag{2.6}$$

$$Holds(Livre(B_2), s_0) \tag{2.7}$$

$$Holds(Livre(B_3), s_0) \tag{2.8}$$

$$Holds(Cor(B_1, Amarelo), s_0) \tag{2.9}$$

$$Holds(Cor(B_2, Amarelo), s_0) \tag{2.10}$$

$$Holds(Cor(B_3, Amarelo), s_0) \tag{2.11}$$

Essas seis fórmulas representam as observações segundo as quais "na situação inicial, os blocos B_1, B_2 e B_3 estão sobre a mesa, livres, isto é, sem outro bloco em cima de cada um deles, e, na mesma situação, a cor dos referidos blocos é amarela".

Nesse cenário, podemos mudar os blocos de lugar, um de cada vez, respeitando as seguintes leis:

$$\begin{aligned} &Holds(Livre(x), s) \wedge Holds(Livre(y), s) \wedge (x \neq Mesa) \rightarrow \\ &\qquad Holds(Sobre(x, y), result(mover(x, y), s) \end{aligned} \tag{2.12}$$

$$\begin{aligned} &Holds(Livre(x), s) \wedge Holds(Livre(y), s) \wedge \\ &Holds(Sobre(x, z), s) \wedge (x \neq y) \wedge (x \neq z) \rightarrow \\ &\qquad Holds(Livre(z), result(mover(x, y), s)) \end{aligned} \tag{2.13}$$

A lei (2.12) representa que, "estando dois blocos livres, então o primeiro estará sobre o segundo após se mover aquele para cima deste".

A lei (2.13), por sua vez, expressa o fato de que, "ao se mover um bloco livre para cima de outro também livre, aquele sobre o qual o primeiro estava também estará livre".

Com essas leis, podemos realizar predições sobre como estará o mundo em situações resultantes de se mover os blocos de lugar. Como exemplo, considerando como situação inicial a disposição dos blocos apresentada na figura 2.1, ao se mover o bloco B_1 para cima de B_2, teremos a situação representada pela figura 2.2.

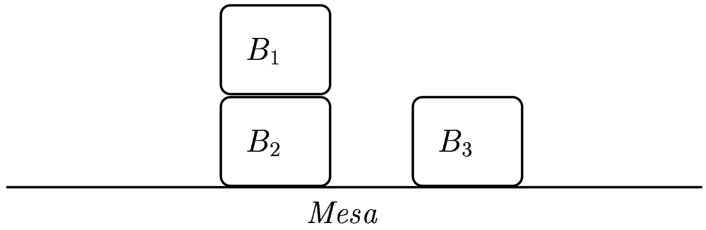

Figura 2.2: Outra configuração possível do mundo dos blocos.

Completando o domínio, temos também a ação de pintar um bloco de vermelho:

$$Holds(Cor(x, Vermelho), result(pintar(x, Vermelho), s)) \qquad (2.14)$$

Tal fórmula expressa o fato de que "após pintar um determinado bloco de vermelho, esta será a sua cor".

Uma vez formalizada a descrição do domínio numa lógica de base, deseja-se poder realizar inferências na base de conhecimento assim obtida. Ou seja, deseja-se poder realizar tarefas de predição, explicação e planificação, tais como definidas na seção 1.2.

Entretanto, é exatamente a partir desse ponto que alguns problemas começam a aparecer. Essa é a motivação das seções seguintes.

2.2 O problema da persistência

Consideremos ainda o cenário representado pelo mundo dos blocos e a teoria de base formalizada pelas fórmulas (2.3)–(2.14). Com essa teoria, tal qual codificada pelas referidas fórmulas, pode-se concluir, por exemplo, que após mover o bloco B_1 para cima de B_2, aquele estará sobre este:

$$(2.3)\text{–}(2.14) \models Holds(Sobre(B_1, B_2), result(mover(B_1, B_2), s_0))$$

Suponhamos, agora, que desejamos saber onde estará o bloco B_3 após ter-se movido B_1 para cima de B_2. Nesse caso, desejamos concluir que ele continuará sobre a mesa, uma vez que o fato de mover B_1 de lugar não interfere na posição de B_3. Entretanto, sem muito esforço, pode-se perceber claramente que, considerando somente as fórmulas (2.3)–(2.14), não é possível concluir

$$(2.3)\text{–}(2.14) \models Holds(Sobre(B_3, Mesa), result(mover(B_1, B_2), s_0)), \qquad (2.15)$$

ou seja, nada se sabe a respeito da posição de B_3 após ter-se movido B_1 para cima de B_2.

Ora, e por que não é possível concluir (2.15)? Porque simplesmente nenhuma informação a respeito da permanência da propriedade $Sobre(B_3, Mesa)$ após a execução de $mover(B_1, B_2)$ está presente na teoria de base e nem dela pode ser derivada. Isso significa, então, que as conclusões que podemos tirar formalmente a partir da teoria considerada são extremamente limitadas, uma vez que só é possível realizar inferências a respeito das mudanças do mundo, sendo que somos obrigados a permanecer neutros quando fazemos questões relacionadas às não-mudanças.

Nesse sentido, não basta enunciarmos apenas os efeitos das ações, mas é preciso também uma maneira de codificar seus "não-efeitos", ou seja, o que não muda em função da execução das mesmas. Sendo assim, para podermos raciocinar corretamente neste mundo dinâmico, faz-se necessária a inclusão dos seguintes axiomas à teoria de base:

$$Holds(Sobre(x,y),s) \rightarrow Holds(Sobre(x,y), result(pintar(x,w),s)) \tag{2.16}$$

$$\begin{gathered} Holds(Sobre(x,y),s) \wedge (x \neq v) \rightarrow \\ Holds(Sobre(x,y), result(pintar(v,w),s)) \end{gathered} \tag{2.17}$$

$$Holds(Livre(x),s) \rightarrow Holds(Livre(x), result(pintar(x,y),s)) \tag{2.18}$$

$$\begin{gathered} Holds(Livre(x),s) \wedge (x \neq y) \rightarrow \\ Holds(Livre(x), result(pintar(y,w),s)) \end{gathered} \tag{2.19}$$

$$\begin{gathered} Holds(Livre(x),s) \wedge (x \neq y) \rightarrow \\ Holds(Livre(x), result(mover(v,y),s)) \end{gathered} \tag{2.20}$$

$$Holds(Cor(x,v),s) \rightarrow Holds(Cor(x,v), result(mover(x,y),s)) \tag{2.21}$$

$$\begin{gathered} Holds(Cor(x,v),s) \wedge (x \neq y) \rightarrow \\ Holds(Cor(x,v), result(mover(y,w),s)) \end{gathered} \tag{2.22}$$

$$\begin{gathered} Holds(Cor(x,v),s) \wedge (x \neq y) \rightarrow \\ Holds(Cor(x,v), result(pintar(y,w),s)) \end{gathered} \tag{2.23}$$

Os axiomas (2.16) e (2.17) codificam, respectivamente, o fato de que "um bloco permanece sobre outro ao pintá-lo ou ao se pintar qualquer outro". Os axiomas (2.18), (2.19) e (2.20) traduzem a idéia de que "um bloco permanece livre ao pintá-lo ou ao se pintar qualquer outro ou ainda ao se mover qualquer bloco para cima de outro". Os axiomas (2.21), (2.22) e (2.23) expressam o fato de que "a cor de um dado bloco não muda se o mesmo ou outro bloco é movido de lugar ou se outro bloco é pintado".

Axiomas da forma de (2.16)–(2.23) são chamados de *axiomas de persistência*[2] e têm a função de explicitar os fatos envolvendo propriedades que *persistem* após a execução de determinadas ações.

Note que, para o cenário acima, tivemos de escrever oito axiomas desse tipo. É exatamente a necessidade de se enunciar tais axiomas na teoria de base que constitui o *problema da persistência*.

Segundo McCarthy e Hayes, para um domínio contendo n fluentes e m ações, o número de axiomas de persistência necessários gira em torno de $n \times m$ [47]. A maioria dos cenários-exemplos analisados na literatura apóia essa conjectura, o que leva a crer que para domínios de interesse prático envolvendo um número muito maior de ações e de fluentes, a presença de tais axiomas chega a constituir um verdadeiro empecilho.

Uma maneira interessante de se tentar evitar de escrever todos os axiomas de persistência necessários para um dado domínio é considerar apenas as leis de efeito das ações e supor que todo o restante permanece inalterado.

2.2.1 A hipótese da inércia

A suposição de que as mudanças que realmente ocorrem são tão somente aquelas explicitamente codificadas, permanecendo o restante inalterado, constitui a chamada *hipótese da inércia*. A idéia intuitiva de tal hipótese é que "normalmente, uma dada ação não altera o estado de um dado objeto".

A hipótese da inércia tem constituído uma das principais ferramentas para a solução do problema da persistência e foi utilizada pela primeira vez por McCarthy no contexto da circunscrição [46]. Ao invés de se escrever todos os axiomas de persistência para um dado domínio, adiciona-se às leis de efeito das ações apenas o seguinte axioma:

$$\neg Afeta(\alpha, f, s) \rightarrow (Holds(f, result(\alpha, s)) \leftrightarrow Holds(f, s)) \tag{2.24}$$

O axioma (2.24) codifica a idéia intuitiva de que "se uma determinada ação α não afeta (não é anormal em relação a) um dado fluente f numa dada situação s, então f permanece inalterado após a execução de α em s".

Sendo assim, em adição ao axioma (2.24), basta que se enunciem fórmulas estabelecendo quais ações afetam quais fluentes no domínio em representação. A não-presença

[2] Na formulação original em inglês, tais axiomas são chamados de *frame axioms*. Muitas traduções em português adotam a expressão "axiomas de quadro".

de uma fórmula explicitando que uma dada ação afeta um dado fluente faz com que se considere que a ação em questão não o afeta[3].

Nesse sentido, a abordagem proposta em [46] consiste em aplicar métodos de circunscrição para minimizar o conjunto de fluentes afetados pelas ações. Com isso, obtém-se uma representação mais econômica do que a que utiliza axiomas de persistência.

Em termos formais, a circunscrição de uma dada teoria A minimizando um predicado P é a fórmula de segunda ordem

$$A \wedge \neg \exists p (A(p) \wedge p < P) \tag{2.25}$$

onde $A(p)$ é a fórmula obtida pela substituição de todas as ocorrências de P em A por p, e $p < P$ representa o fato de que a extensão de p, i.e. o conjunto de objetos sobre os quais p é verdadeiro, é um subconjunto próprio da de P. Intuitivamente, a fórmula (2.25) estabelece que a extensão de um dado predicado deve ser mínima com respeito aos modelos possíveis da teoria considerada.

Assim, a maneira com que McCarthy formaliza a hipótese da inércia em sua proposta consiste no enunciado de uma fórmula da forma de (2.25) com P substituído por $Afeta$.

Entretanto, apesar de a hipótese da inércia ser uma idéia intuitivamente válida, a abordagem apresentada por McCarthy para sua formalização não soluciona completamente o problema da persistência. O ponto crítico reside no fato de sua proposta não se comportar bem em domínios envolvendo ações sem efeito, como por exemplo o cenário do tiro em Yale [27], que analisaremos a seguir.

2.2.2 O cenário do tiro em Yale

Suponhamos uma situação na qual se tenha uma vítima, normalmente um peru[4], o qual pode estar vivo, representado por $Vivo$, ou morto, representado por $\neg Vivo$. Suponhamos também que temos uma arma, que pode estar carregada ou descarregada, fatos estes representados, respectivamente, por $Carregada$ e $\neg Carregada$.

Suponha ainda que podemos executar a ação de carregar a arma, representada por *carregar* e que tem como efeito tornar a arma carregada, e a ação de atirar com a referida arma, representada por *atirar*, a qual tem como efeito matar o peru, desde que a arma esteja carregada.

Além dessas ações, temos ainda a ação de esperar um instante de tempo, representada por *esperar*, e que, intuitivamente, não deve produzir efeito algum.

[3]Vale a pena destacar que tal noção em muito lembra o processo de *negação por falha* fundamentado na *hipótese do mundo fechado* [11].

[4]No cenário original em [27], a vítima é um ser humano. Entretanto, nos trabalhos subseqüentes a comunidade passou a adotar um peru, justamente por ser mais politicamente correto.

A modelagem de tal cenário no cálculo de situações formalizando a hipótese da inércia é dada pelas fórmulas a seguir:

$$Holds(Carregada, result(carregar, s)) \tag{2.26}$$

$$Holds(Carregada, s) \rightarrow \neg Holds(Vivo, result(atirar, s)) \tag{2.27}$$

$$\neg Afeta(\alpha, f, s) \rightarrow (Holds(f, result(\alpha, s)) \leftrightarrow Holds(f, s)) \tag{2.28}$$

$$Holds(Vivo, s_0) \tag{2.29}$$

$$\neg Holds(Carregada, s_0) \tag{2.30}$$

A lei de efeito (2.26) representa o fato de que "após se executar a ação de carregar a arma, a mesma estará carregada". A lei de efeito (2.27) expressa que "se a arma estiver carregada, então após atirar, o peru estará morto". O axioma (2.28), por seu turno, codifica a lei da inércia segundo a qual "propriedades não explicitamente afetadas permanecem inalteradas". Já as fórmulas (2.29) e (2.30) constituem, respectivamente, as observações feitas na situação inicial de que o peru está vivo e a arma descarregada.

A partir dessa descrição do domínio, deseja-se saber qual o estado do mundo após a execução em seqüência das ações de carregar a arma, esperar um instante e em seguida atirar no peru. Ou seja, queremos saber quais as propriedades que serão válidas na situação dada por $result(atirar, result(esperar, result(carregar, s_0)))$. Em particular, deseja-se poder concluir que, após tal seqüência de ações, o peru estará morto, ou seja:

$$\neg Holds(Vivo, result(atirar, result(esperar, result(carregar, s_0)))). \tag{2.31}$$

Infelizmente, com a utilização da hipótese da inércia em combinação com métodos de circunscrição, a fórmula (2.31) não pode ser inferida a partir da teoria de base considerada. Isso se deve à existência de *modelos anômalos* resultantes do processo de minimização nos quais a anormalidade de certas ações acaba sendo compensada pela de outras [27].

Para esse exemplo em particular, existem modelos anômalos nos quais a ação *esperar* é interpretada como anormal em relação ao fluente *Carregada*, tendo o efeito de descarregar a arma. Em função disso, a ação *atirar* deixa de ser anormal em relação ao fluente *Vivo*, mantendo-o inalterado, uma vez que a pré-condição para que produza efeito, isto é, a arma estar carregada, já não vale mais. A figura 2.3 ilustra essa circunstância.

Observe, na referida figura, as transições entre os estados w_0 e w_1. Note que é possível carregar a arma, causando o efeito de carregá-la, esperar, produzindo o efeito de descarregá-la, e em seguida atirar, sem produzir efeito algum. Apesar de completamente não-intuitivo, esse modelo é perfeitamente possível do ponto de vista da lógica formal e não é restringido pelo método proposto em [46].

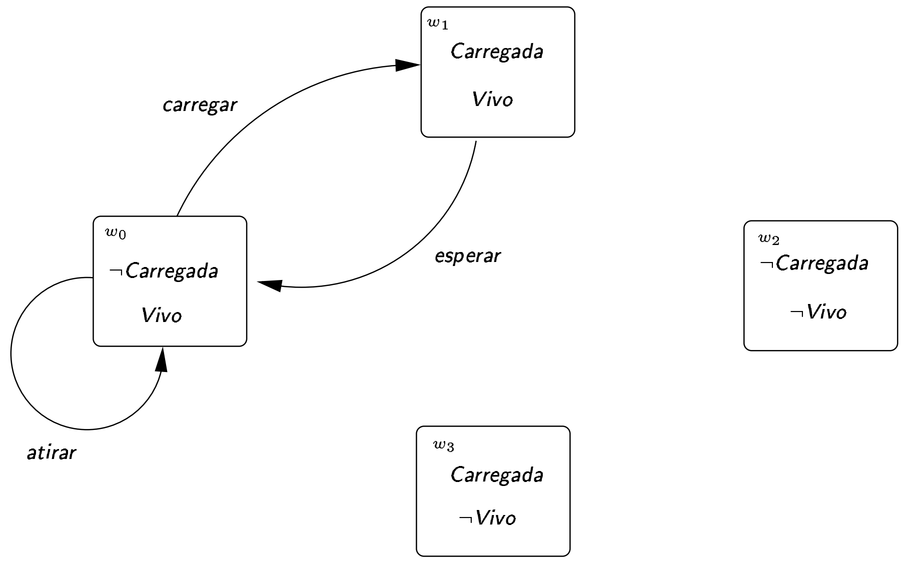

Figura 2.3: Um dos modelos anômalos do cenário do tiro em Yale (por questões de clareza, nem todas as transições possíveis foram apresentadas).

A descoberta do cenário do tiro em Yale constituiu um verdadeiro marco na história da investigação em raciocínio sobre ações, pois invalidou uma série de soluções até então propostas. Desde então, esse cenário vem sendo amplamente utilizado como cenário clássico para se abordar o problema da persistência.

2.3 O problema da ramificação

Consideremos, agora, um cenário bastante simples envolvendo uma maleta automática com dois fechos [37]. $ParaCima_1$ e $ParaCima_2$ representam, respectivamente, os fatos de que o primeiro fecho e o segundo estão abertos. $Aberta$ expressa o fato de a maleta estar aberta. Podemos ainda executar as ações $levantar_1$ e $levantar_2$, que têm o efeito de abrir o primeiro e o segundo fechos, respectivamente.

Nesse cenário, deseja-se formalizar o fato de que toda vez em que se abrir um dos fechos, estando o outro também aberto, a maleta se abre automaticamente. Em outros termos, deseja-se poder captar os eventuais efeitos indiretos causados pela ação de abrir um único fecho. Tal cenário constitui uma instância do *problema da ramificação*.

Utilizando o cálculo de situações como lógica de base e esquecendo por hora o problema da persistência, teríamos, para esse cenário, a seguinte formalização:

$$\neg Holds(ParaCima_1, s) \leftrightarrow Holds(ParaCima_1, result(levantar_1, s)) \tag{2.32}$$

$$\neg Holds(ParaCima_2, s) \leftrightarrow Holds(ParaCima_2, result(levantar_2, s)) \tag{2.33}$$

$$Holds(ParaCima_1, s) \wedge Holds(ParaCima_2, s) \leftrightarrow Holds(Aberta, s) \qquad (2.34)$$

$$\neg Holds(ParaCima_1, s_0) \qquad (2.35)$$

$$Holds(ParaCima_2, s_0) \qquad (2.36)$$

Os axiomas (2.32) e (2.33) codificam os efeitos diretos das ações $levantar_1$ e $levantar_2$, respectivamente. A lei de domínio (2.34) expressa a restrição segundo a qual "toda vez em que os dois fechos estiverem abertos, então a maleta se abre". As observações (2.35) e (2.36) representam a situação inicial na qual um dos fechos está fechado e o outro aberto.

Observe que a partir da restrição (2.34), podemos concluir que na situação inicial a maleta está fechada.

Numa primeira instância, abstraindo o problema da persistência, parece não haver dificuldade alguma em se determinar os efeitos indiretos da ação $levantar_1$, face à restrição de domínio (2.34), a qual permite a dedução dos mesmos.

De fato, um consenso na comunidade de raciocínio sobre ações com relação ao problema da ramificação sempre foi o de que a solução para o mesmo consistiria no estabelecimento de um conjunto completo de restrições de domínio, de maneira que os efeitos indiretos poderiam ser deduzidos facilmente por meio delas.

Entretanto, a simples utilização de leis de domínio não é suficiente para se obter as ramificações (todas as conseqüências) de determinadas ações. Considere, por exemplo, a situação inicial acima codificada. Neste caso, ao se executar a ação $levantar_1$, a restrição de domínio (2.34) acaba assumindo um papel completamente ambíguo em relação aos resultados da execução da referida ação: ou a maleta se abre ou então o segundo fecho acaba misteriosamente se fechando, mantendo a maleta fechada. Isso acontece devido ao fato de a partir da restrição de domínio (2.34) podermos deduzir a seguinte fórmula:

$$Holds(ParaCima_1, s) \wedge \neg Holds(Aberta, s) \rightarrow \neg Holds(ParaCima_2, s) \qquad (2.37)$$

Esse problema também pode ser considerado como resultante da existência de modelos anômalos, tal qual ilustrado na figura 2.4. Observe, na referida figura, que tanto o estado w_1 como o w_2 satisfazem a restrição de domínio (2.34).

Em função disso, percebe-se que a simples codificação de leis de domínio não basta para resolver o problema da ramificação. Sendo assim, deve-se buscar uma maneira de sobrepujar certas "propriedades más" da implicação material [8], como a derivação da fórmula (2.37) a partir de (2.34). Isso leva a crer que algum mecanismo além da lógica formal se faz necessário para atingir esse objetivo.

É justamente nesse ponto que se fundamenta a motivação para o uso de *abordagens causais*, ou seja, maneiras de se representar uma noção de causalidade envolvendo as

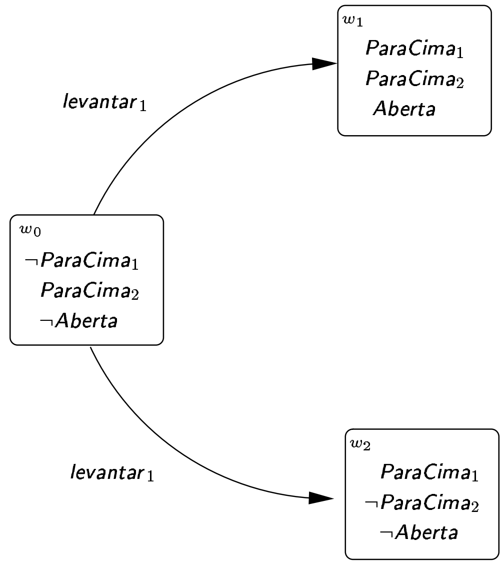

Figura 2.4: Modelo anômalo da maleta automática de Lin [37] para a ação $levantar_1$.

mudanças do mundo. Nesse sentido, uma forma de atacar o problema da ramificação consiste na formalização de que mudanças devem ser explicitamente causadas, e não fruto de alterações espontâneas. Esse ponto constitui uma das principais motivações do presente trabalho e teremos a oportunidade de retornar a ele na seqüência.

2.4 O problema da qualificação

Até agora, falamos de ações, seus efeitos diretos e indiretos, mudanças e não-mudanças do mundo, mas em que momento foi explicitado que cada uma das ações consideradas é realmente executável? O que garante que $mover(B_1, B_2)$, *atirar* e $levantar_1$ podem ser executadas em seus respectivos cenários? Tais questões estão ainda muito longe de obter uma resposta e constituem o chamado *problema da qualificação*.

Basicamente, esse problema consiste em encontrar uma maneira de se poder deduzir a possibilidade de execução de uma determinada ação sem a necessidade de explicitar todas as pré-condições necessárias para tal. Entretanto, nenhuma maneira alternativa a essa foi encontrada até hoje.

Considerando novamente o mundo dos blocos, uma maneira bastante rudimentar de explicitar a possibilidade de execução das ações é enunciando *leis de executabilidade* como:

$$Poss(mover(x, y), s)$$

$$Poss(pintar(x, y), s)$$

Tais fórmulas expressam, respectivamente, os fatos de que "é possível mover um bloco para cima de outro" e de que "é possível pintar um determinado bloco".

Mas, nesse caso, o que dizer se acabarmos descobrindo que um dos blocos está pregado na mesa? Ou então que a tinta repentinamente acabou?

Essa é exatamente a essência do problema da qualificação: como podemos ter certeza de que as pré-condições para a execução de uma ação codificadas na base de conhecimento são realmente *todas* as pré-condições que existem? Da mesma forma que podemos tomar conhecimento da existência de novas ações e novos fluentes, também podemos tomar conhecimento de novas pré-condições para a execução de uma ação. Isso faz com que, de uma hora para a outra, uma determinada ação possa não ser mais executável, e constitui interesse para um ser inteligente deduzir o seu porquê.

Inúmeras evidências apontam que uma solução para esse problema talvez deva ser fundamentada em abordagens não-monotônicas [50, 46] ou em mecanismos de revisão de crenças [1, 19]. Entretanto, até o presente momento, nenhuma tentativa concreta em termos de solução para o problema da qualificação foi empreendida.

Ao que tudo indica, dentre os três principais problemas em raciocínio sobre ações, o da qualificação é o de maior complexidade e o de mais difícil solução. Tal complexidade se deve, em parte, ao fato de ainda não estar bem definido o que são efetivamente todas as pré-condições relevantes de uma ação e até que ponto é viável uma total onisciência das mesmas. Sendo assim, no presente trabalho fazemos referências a qualificações apenas em situações específicas, não tendo por objetivo propor uma solução final para tal problema.

2.5 Critérios para uma solução satisfatória

Ao longo do presente trabalho, teremos a oportunidade de analisar as principais abordagens propostas para raciocinar sobre ações e compará-las entre si, ilustrando tanto seus pontos fortes como os problemas que apresentam.

Para determinar se um dado formalismo proposto para raciocinar sobre ações pode ou não ser considerado como uma *solução satisfatória* para os problemas da área, seguiremos ao longo deste trabalho os seguintes critérios, compilados em [63, seção 1.6] e em grande parte aceitos na comunidade:

1. Para ser satisfatória, uma proposta de solução deve permitir uma representação econômica do domínio em consideração, no sentido de que o responsável pela codificação do mesmo tenha de escrever um número de fórmulas proporcional à descrição do problema em língua natural. Um formalismo que não atenda a essa restrição não pode ser considerado como aceitável.

2. Tal solução deve possibilitar também uma grande flexibilidade em relação às classes de problemas consideradas. Isso significa que um formalismo proposto precisaria ser capaz de tratar ações mais complexas envolvendo efeitos indiretos e não-determinísticos. Segundo Shanahan, seria também desejável que uma solução proposta pudesse tratar ações concorrentes e mudança contínua [63], entretanto no presente trabalho não abordaremos estas classes de ações.

3. A solução em questão deve também apresentar um alto grau de modularidade no tocante a futuras modificações. Isso quer dizer que se uma nova informação é fornecida ao projetista do sistema, obrigando-o a alterar a descrição do domínio, então o trabalho necessário para realizar tal mudança deve ter uma complexidade proporcional à da nova informação considerada. Na realidade, esse critério é bastante discutível, uma vez que há muitas abordagens que não são nem um pouco modulares e que se comportam muito bem face aos principais problemas abordados.

4. Tal solução deve ainda possibilitar a realização das tarefas fundamentais de predição, explicação e planificação, definidas na seção 1.2. Um formalismo que seja restrito a apenas uma ou duas de tais tarefas não constitui, portanto, uma solução satisfatória.

5. Por fim, é de fundamental importância que a solução proposta possua métodos de inferência adequados, completos e decidíveis que possam dotar de um caráter prático o formalismo apresentado.

A partir do próximo capítulo, empreenderemos uma análise das principais abordagens causais presentes na literatura que, acreditamos, resumem bem o estado da arte em raciocínio sobre ações. Nosso objetivo será investigar como tais abordagens formalizam uma noção de causalidade para tratar os problemas da persistência e da ramificação, sendo que utilizaremos os critérios acima definidos como base para uma avaliação crítica.

Parte II

Abordagens Clássicas

Capítulo 3

Minimização da causalidade

> *Quando se exclui de uma situação todas as impossibilidades, o que sobrar, por mais improvável que pareça, deverá ser a verdade.*
>
> — Arthur Conan Doyle

Conforme vimos no capítulo anterior, a simples utilização de restrições de domínio, que se referem apenas aos valores de verdade das proposições envolvidas, não é suficiente para se obter uma solução em conjunto para os problemas da persistência e da ramificação, sendo, portanto, necessário o uso de uma noção de causalidade para representar os efeitos indiretos das ações.

Diversas abordagens causais não-monotônicas fundamentadas no cálculo de situações foram propostas na literatura, sendo que neste capítulo apresentaremos a abordagem de Lin [37], que resume bem a problemática envolvida em todas elas.

No referido trabalho, efetua-se uma extensão do cálculo de situações para comportar uma noção causal mediante a definição de um predicado que estabelece que as mudanças de determinadas propriedades são *causadas* por certos fatores em determinadas circunstâncias. Com o intuito de restringir os modelos àqueles em que realmente se tem mudanças intuitivas, reduz-se ao máximo o conjunto de efeitos causados através de uma política de minimização circunscritiva.

3.1 Fundamentos

Em essência, a abordagem aqui apresentada consiste na definição de um cálculo de situações (ver seção 2.1) acrescido de um predicado ternário $Caused$, o qual é utilizado para representar a semântica causal proposta. Uma vez definida a teoria, o passo seguinte reside na utilização de uma variante de métodos de circunscrição [45, 46] para promover, assim, a minimização da mesma.

O novo predicado $Caused(p, v, s)$ definido é interpretado como sendo verdadeiro se o fluente denotado por p é levado a receber o valor de verdade v numa dada situação s. p, nesse caso, é entendido como tendo sido causado por alguma coisa qualquer, não especificada, que pode ser tanto uma ação como a presença de um outro fluente.

O ponto chave desta abordagem, com relação à definição do predicado *Caused*, consiste no estabelecimento da seguinte distinção entre efeitos causados por ações e efeitos causados por propriedades verificadas: se uma determinada propriedade tem seu valor de verdade alterado por uma certa ação, ou seja, se ela é causada por uma ação, então tem-se um *efeito direto* da ação em questão; já se a mesma alterou seu valor de verdade em função do de outra propriedade presente no domínio, tem-se um *efeito indireto* da ação que originou esta última, ou seja, a primeira propriedade é dita como tendo sido causada pela outra.

Nesse sentido, a base semântica da proposta aqui analisada reside na suposição de que o que provoca efeitos diretos são sempre (e somente) ações, enquanto que o que produz efeitos indiretos são sempre (e somente) efeitos anteriormente produzidos.

Sendo assim, o raciocínio utilizado na determinação do conjunto de efeitos diretos e indiretos resultantes da execução de uma certa ação segue esta linha guia: uma dada ação produz um certo conjunto de alterações nos valores de verdade das propriedades consideradas; tais alterações constituem os chamados efeitos diretos da execução da ação; os novos valores produzidos são, assim, usados na determinação de que valores de verdade as demais propriedades deverão ter na presença daquelas anteriormente geradas.

Com essa interpretação causal, tem-se uma maneira diferente de enxergar a produção de efeitos indiretos, focando-se mais os efeitos das ações como geradores de novos efeitos do que a própria ação como causadora primária de todos os efeitos produzidos.

Como um exemplo dessa visão, tem-se o cenário da maleta automática, analisado na seção 2.3. Nesse cenário, a interpretação causal aqui introduzida não é de que a ação de levantar o primeiro fecho, estando o segundo aberto, causa o efeito indireto de abrir a maleta, mas sim de que o *fato* de os dois fechos estarem abertos é que causa a abertura instantânea da mesma. No decorrer do capítulo, retornaremos ainda a esse ponto.

A seguir, apresentamos a versão do cálculo de situações utilizada como formalismo de base do método aqui analisado.

3.2 Lógica de base

A axiomática utilizada nesta abordagem considera como básicos os seguintes axiomas:

$$Caused(p, verdadeiro, s) \rightarrow Holds(p, s) \tag{3.1}$$

$$Caused(p, falso, s) \rightarrow \neg Holds(p, s) \tag{3.2}$$

segundo os quais, "se um determinado fluente p assume o valor de verdade *verdadeiro* (respectivamente *falso*) numa dada situação s, então ele passa a valer (respectivamente não valer) nessa mesma situação s".

A fórmula

$$verdadeiro \neq falso \wedge \forall v(v = verdadeiro \vee v = falso) \tag{3.3}$$

representa a hipótese de unicidade de nomes e o princípio do terceiro excluído[1].

As fórmulas

$$F_i(x) \neq F_j(x), \qquad \text{para } i \text{ e } j \text{ diferentes,}$$
$$F_i(x) = F_i(y) \rightarrow x = y$$

onde F_i e F_j são nomes de fluentes, $0 \leq i, j \leq n$, para n nomes de fluentes, representa a hipótese de unicidade de nomes para os nomes de fluentes. Em outras palavras, cada fluente é designado por um e somente um nome. A mesma hipótese é levada em conta com relação aos nomes de ações.

As fórmulas

$$s \neq result(\alpha, s) \text{ e} \tag{3.4}$$

$$result(\alpha_1, s_1) = result(\alpha_2, s_2) \rightarrow (\alpha_1 = \alpha_2 \wedge s_1 = s_2) \tag{3.5}$$

constituem a hipótese de unicidade de nomes para situações. Em outros termos, "uma situação não é resultado de se executar uma ação nessa mesma situação", e "se duas situações resultantes da aplicação de duas ações são iguais, então as ações executadas bem como a situação original são as mesmas para ambas".

Na versão de cálculo de situações desta abordagem, utiliza-se ainda o conceito de *fórmula simples de estado sobre uma situação s*, que, em essência, segue a mesma definição de *fórmula simples* [56]. Basicamente, uma fórmula simples de estado sobre uma dada situação s é toda e qualquer fórmula $\Phi(s)$ que não mencione *Poss*, *Caused* ou qualquer outra situação que não s.

Em outros termos, uma fórmula simples de estado é uma fórmula que não se refere à possibilidade de se executar alguma ação em s, ao fato de certos efeitos terem ou não sido causados em s e nem a situações resultantes da execução de ações em s. Em síntese, $\Phi(s)$ deve denotar apenas e tão somente o que é verdadeiro em s.

Exemplos de fórmulas simples de estado são

$$Fecho(L_1) \wedge L_1 \neq L_2,$$

$$Holds(ParaCima(x), s) \wedge \alpha = levantar(x).$$

[1]Do latim *tertium non datur*: todo e qualquer valor de verdade ou é *verdadeiro* ou é *falso*.

Tais fórmulas expressam, respectivamente, que "L_1 é um fecho e é diferente de L_2" e que "o fecho denotado por x está aberto na situação denotada por s e α denota a ação de *levantar* o objeto denotado por x".

Já a fórmula

$$Holds(ParaCima(x), result(\alpha, s))$$

não é uma fórmula simples de estado, pois faz referência à situação resultante de se aplicar a ação denotada por α em s, que, pelo axioma (3.4), é diferente de s.

A seguir, analisaremos a maneira como esse formalismo de base é empregado para raciocinar sobre ações.

3.3 Método empregado

A abordagem de minimização da causalidade consiste numa seqüência de sete passos. Os passos 1–4 apresentam os procedimentos para a construção de uma teoria de base, enquanto que os passos restantes aplicam raciocínio não-monotônico à teoria recém obtida para se abordar os problemas da persistência, da ramificação e da qualificação.

Abaixo apresentamos em maiores detalhes o procedimento acima citado, porém com alguns comentários que têm por objetivo tornar mais clara a exposição de cada passo bem como justificar o emprego de certas tarefas realizadas.

Passo 1. Para cada ação α, formalizar todos os seus *efeitos diretos* através do uso de axiomas da forma:

$$Poss(\alpha, s) \rightarrow (\Phi(s) \rightarrow Caused(F(y), v, result(\alpha, s))), \tag{3.6}$$

onde $F(y)$ é um fluente e $\Phi(s)$ é uma fórmula simples de estado sobre s.

Axiomas dessa natureza querem dizer que, "se é possível executar a ação α na situação s, então, no contexto representado pela fórmula $\Phi(s)$, a propriedade F, no objeto denotado por y, será causada a receber o valor de verdade v na situação resultante $result(\alpha, s)$". Com fórmulas desse tipo, obtém-se uma maneira de representar todos os efeitos diretos de cada ação.

Passo 2. Para cada ação α, formalizar as *qualificações explícitas* de α mediante o seguinte tipo de axiomas:

$$Poss(\alpha, s) \rightarrow \Phi(s), \tag{3.7}$$

com $\Phi(s)$ uma fórmula simples de estado sobre s.

Axiomas dessa forma representam o fato de que $\Phi(s)$ (as fórmulas que são verdadeiras em s) é uma condição necessária para que a ação α seja executável (em s).

Passo 3. Formalizar todas as *regras causais*, ou seja, os enunciados que serão usados para se determinar os efeitos indiretos das ações, através de axiomas da forma:

$$\Phi(s) \land Caused(p_1, v_1, s) \land \cdots \land Caused(p_n, v_n, s) \rightarrow Caused(F(x), v, s), \tag{3.8}$$

onde $F(x)$ é um fluente e $\Phi(s)$ uma fórmula simples de estado sobre s.

Com fórmulas desse tipo, deduz-se quais são os efeitos originados a partir do estado do mundo na situação considerada ($\Phi(s)$) bem como a partir de outros efeitos originados na mesma situação (cada um dos $Caused(p_i, v_i, s)$, para $0 \leq i \leq n$). Note que tais enunciados envolvendo o predicado *Caused* representam apenas mudança potencial[2], de maneira que na seqüência se procederá à minimização do mesmo através do uso de uma política de circunscrição.

Passo 4. Completar a base de conhecimento acerca do domínio mediante axiomas da forma:

$$\forall s C(s) \tag{3.9}$$

onde $C(s)$ é uma fórmula simples de estado sobre s. Tais axiomas constituem, na realidade, as restrições de domínio[3].

Com isso, obteve-se até aqui uma teoria de base que, juntamente com os axiomas da seção 3.2, constituem a representação do domínio de interesse. Uma vez de posse da teoria então construída, o objetivo agora consiste em aplicar raciocínio não-monotônico a fim de obter uma solução para os problemas da persistência, da ramificação e da qualificação. Os passos 5–7 ilustram o processo.

Passo 5. Obter para cada fluente um *axioma de causa* da seguinte forma:

$$Caused(F(x), v, s) \leftrightarrow \Psi, \tag{3.10}$$

[2]Na realidade, a abordagem aqui apresentada baseia-se numa noção causal forte, isto é, que realmente *causa* mudança, ao contrário de outras que possuem uma noção de causalidade que apenas *permite* mudança [65, 7]. Essa diferença é um tanto sutil, sendo que teremos a oportunidade de melhor analisá-la nos próximos capítulos. Dizer que as regras causais representam mudança potencial significa que, do ponto de vista lógico, dão margem a modelos nos quais podem ocorrer mudanças espontâneas não-intuitivas. Daí a necessidade de sujeitar o predicado *Caused* a uma política de circunscrição, conforme será ilustrado na seqüência.

[3]C vem do inglês *constraint* (restrição).

onde Ψ deve ser uma fórmula na qual não conste o predicado *Caused*.

Os axiomas de causa são obtidos através da circunscrição do predicado *Caused* na teoria obtida nos passos 1–4, com todos os demais predicados fixos [36]. O grande trunfo deste passo reside no fato de, se a teoria em questão for *estratificada* [37], ser possível calcular sua circunscrição mediante a compleção de predicados de Clark [11].

Formalmente, uma teoria é dita estratificada se ela não possuir fluentes $F_0, F_1, \ldots, F_n$ tais que $F_0 \rightarrow F_1 \rightarrow \ldots \rightarrow F_n \rightarrow F_0$, onde para quaisquer dois fluentes F_i, F_j, $0 \leq i, j \leq n$, $F_j \rightarrow F_i$ (F_i depende de F_j) se houver uma regra causal (como, por exemplo, (3.8)) na qual F_i apareça no lado direito da regra e F_j no lado esquerdo da mesma[4].

Sejam A uma fórmula e P um predicado tais que $A \rightarrow P$. A compleção do predicado P consiste basicamente na substituição da implicação $A \rightarrow P$ pela equivalência $A \leftrightarrow P$, desde que P não apareça em A [11, 36].

Passo 6. Uma vez obtidos, para cada fluente, os axiomas de causa da forma de (3.10), o objetivo deste passo é calcular, novamente para cada fluente, um *axioma de estado sucessor* [56]. Axiomas de estado sucessor são uma maneira sucinta de combinar todos os axiomas de efeito com todos os axiomas de fecho de explicação (ver capítulo 5) para um dado fluente em uma única fórmula [63] e apresentam a seguinte forma:

$$\begin{aligned} Poss(\alpha, s) \rightarrow \quad & Holds(F(x), result(\alpha, s)) \leftrightarrow \\ & (\Phi(s) \vee (Holds(F(x), s) \wedge \Phi'(s))), \end{aligned} \tag{3.11}$$

onde $\Phi(s)$ e $\Phi'(s)$ são fórmulas simples de estado sobre s.

Lin [37] afirma que *geralmente* pode-se obter um axioma de estado sucessor como (3.11), para cada fluente, se a teoria considerada for estratificada. Segundo ele, tendo-se obtido para cada fluente um axioma de estado sucessor, obtém-se a solução para os problemas da persistência e da ramificação. Não entraremos em detalhes quanto a isso agora, pois este é um ponto importante ao qual retornaremos no final do capítulo.

Passo 7. Supondo-se que para cada fluente foi calculado um axioma de estado sucessor nos moldes de (3.11), deseja-se, neste passo, obter para cada ação α um *axioma de pré-condição da ação* [56] da forma:

$$Poss(\alpha, s) \leftrightarrow \Phi(s) \tag{3.12}$$

[4]Um exemplo de dependência entre fluentes envolvendo regras causais é a dependência entre *ParaCima* e *ParaBaixo* em função das regras $Caused(ParaCima, verdadeiro, s) \leftrightarrow Caused(ParaBaixo, falso, s)$ e $Caused(ParaCima, falso, s) \leftrightarrow Caused(ParaBaixo, verdadeiro, s)$. Neste caso, temos como resultado a dependência $ParaCima \rightarrow ParaBaixo \rightarrow ParaCima$, de modo que uma teoria que contivesse tais regras e fluentes não seria estratificada. Sendo assim, em teorias desse tipo, a compleção de predicados não é aplicável. As conseqüências resultantes de situações como essa serão discutidas ao final do capítulo.

onde $\Phi(s)$ é uma fórmula que se refere somente aos valores de verdade dos fluentes em s.

Em outras palavras, deseja-se que as pré-condições de cada ação possam ser expressas em termos dos valores de verdade dos fluentes válidos na situação atual. Isso é obtido através da maximização do predicado $Poss(\alpha, s)$ com $Holds(p, s)$ fixo, variando *Caused* e $Holds(p, s') \wedge s \neq s'$ [36]. Assim, obtém-se um modelo no qual não se pode aumentar a extensão de $Poss(\alpha, s)$ seja através de uma mudança na interpretação de *Caused* seja por uma mudança nos valores de verdade dos fluentes em situações que não s.

A maneira como tal maximização é realizada consiste na aplicação da técnica de *regressão* [56] seguida da compleção de predicados de Clark [11].

Em linhas gerais, a regressão é uma série de manipulações sintáticas empreendidas em cada fórmula de uma dada teoria cuja finalidade é substituir todas as ocorrências de fórmulas contendo a função *result* por fórmulas equivalentes.

A razão de se aplicar a regressão e em seguida a compleção de predicados é que, com isso, obtém-se os axiomas de pré-condição de ações da forma de (3.12) sem a necessidade de se utilizar de maneira explícita uma política de circunscrição.

Como resultado fundamental do procedimento acima definido, tem-se o seguinte teorema [37]:

Teorema 3.3.1 Se uma teoria T obtida pelos passos 1–4 é *estratificada*, então existe um procedimento para obter um *axioma de estado sucessor* para cada fluente e um *axioma de pré-condição da ação* para cada ação. Tal procedimento consiste em simples manipulações sintáticas e é adequado com respeito à semântica não-monotônica apresentada. ∎

3.4 Cenário-exemplo

A fim de bem ilustrar a aplicação do método de minimização da causalidade, apresentamos nesta seção um exemplo de sua utilização.

Exemplo 3.4.1 (O peru que caminha [2]) Este cenário consiste de uma extensão do cenário do tiro em Yale [27] para comportar ramificações. Considere um agente, denotado por um peru, que pode realizar as ações de começar a caminhar, representada na formulação em cálculo de situações por *começar-a-andar*, e parar de caminhar, representada por *parar-de-andar*. Suponha ainda a ação externa de atirar, representada por *atirar*. Na situação inicial, o peru está vivo e caminhando.

Nesse sentido, deseja-se poder inferir o efeito indireto de que o peru não estará mais caminhando após ter-lhe sido executada a ação de atirar, bem como a qualificação implícita (axioma implícito sobre pré-condições de uma ação) [23, 40] de que, para poder começar a caminhar, o peru precisa estar vivo.

Passo 1. Para este cenário, os axiomas que representam os efeitos diretos das ações definidas são os seguintes:

$$\begin{aligned}
&Poss(\textit{começar-a-andar}, s) \rightarrow \\
&\qquad Caused(\textit{Caminhando}, \textit{verdadeiro}, \textit{result}(\textit{começar-a-andar}, s)), \\
&Poss(\textit{parar-de-andar}, s) \rightarrow \\
&\qquad Caused(\textit{Caminhando}, \textit{falso}, \textit{result}(\textit{parar-de-andar}, s)), \\
&\quad Poss(\textit{atirar}, s) \rightarrow Caused(\textit{Morto}, \textit{verdadeiro}, \textit{result}(\textit{atirar}, s)).
\end{aligned} \tag{3.13}$$

Passo 2. Qualificações explícitas:

$$Poss(\textit{começar-a-andar}, s) \rightarrow \neg Holds(\textit{Caminhando}, s),$$
$$Poss(\textit{parar-de-andar}, s) \rightarrow Holds(\textit{Caminhando}, s).$$

ou seja, as únicas qualificações que se sabe é que para o peru poder começar a (respectivamente parar de) caminhar é preciso que ele não esteja (respectivamente esteja) caminhando.

Passo 3. Determinação da regra causal:

$$Holds(\textit{Morto}, s) \rightarrow Caused(\textit{Caminhando}, \textit{falso}, s), \tag{3.14}$$

a qual determina que o fato de o peru estar morto numa dada situação s automaticamente causa o fato de o fluente *Caminhando* tornar-se falso em s.

Note que (3.14), em combinação com (3.13) e com os axiomas (3.1) e (3.2), permite a dedução do seguinte efeito indireto de *atirar*:

$$Poss(\textit{atirar}, s) \rightarrow \neg Holds(\textit{Caminhando}, \textit{result}(\textit{atirar}, s)). \tag{3.15}$$

Passo 4. Para este exemplo, não há outras restrições de domínio.

Passo 5. A teoria obtida nos passos anteriores é estratificada. Neste sentido, mediante a aplicação da compleção de Clark [11] ao predicado *Caused*, obtém-se os seguintes axiomas de causa:

$$\begin{aligned}
Caused(\textit{Morto}, v, s) \leftrightarrow \quad & v = \textit{verdadeiro} \wedge \\
& \exists s'[s = \textit{result}(\textit{atirar}, s') \wedge Poss(\textit{atirar}, s')]
\end{aligned} \tag{3.16}$$

$$\begin{aligned}
&Caused(Caminhando, v, s) \leftrightarrow \\
&\quad v = falso \wedge Holds(Morto, s) \vee \\
&\quad v = falso \wedge \exists s'[s = result(parar\text{-}de\text{-}andar, s') \wedge \\
&\qquad\qquad Poss(parar\text{-}de\text{-}andar, s')] \vee \\
&\quad v = verdadeiro \wedge \exists s'[s = result(começar\text{-}a\text{-}andar, s') \wedge \\
&\qquad\qquad Poss(começar\text{-}a\text{-}andar, s')]
\end{aligned} \tag{3.17}$$

O axioma (3.16) expressa que "o peru estará morto após se executar *atirar* numa situação em que tal ação seja executável". O axioma de causa (3.17) traduz que "o peru não estará mais caminhando após ter parado de andar ou após ter sido morto, ou então estará caminhando após ter começado a andar".

Passo 6. Com os resultados obtidos até então, constroem-se os seguintes axiomas de estado sucessor para os fluentes *Morto* e *Caminhando*, respectivamente:

$$Poss(\alpha, s) \rightarrow (Holds(Morto, result(\alpha, s)) \leftrightarrow ((\alpha = atirar) \vee Holds(Morto, s))),$$

$$\begin{aligned}
Poss(\alpha, s) \rightarrow \quad & Holds(Caminhando, result(\alpha, s)) \leftrightarrow \\
& \alpha = começar\text{-}a\text{-}andar \vee Holds(Caminhando, s) \wedge \\
& \neg Holds(Morto, s) \wedge \alpha \neq parar\text{-}de\text{-}andar \wedge \alpha \neq atirar
\end{aligned}$$

O primeiro axioma de estado sucessor enuncia que "se é possível executar uma determinada ação α numa dada situação s, então o peru estará morto na situação resultante se e somente se tal ação for *atirar* ou então o peru já estiver morto em s". O segundo, por sua vez, denota que "se é possível executar uma certa ação α numa situação s, então o peru estará caminhando na situação resultante se e somente se tal ação é *começar-a-andar* ou já estava caminhando em s e na mesma situação não estava morto e a ação em questão não é *parar-de-andar* nem *atirar*". Esses axiomas de estado sucessor permitem uma solução para os problemas da persistência e da ramificação.

Passo 7. Após algumas manipulações sintáticas nos axiomas causais obtidos anteriormente e aplicando-se a técnica de regressão nos fluentes *Morto* e *Caminhando* seguida da compleção de predicados de Clark, obtém-se:

$$Poss(começar\text{-}a\text{-}andar, s) \leftrightarrow \neg Holds(Caminhando, s) \wedge \neg Holds(Morto, s), \tag{3.18}$$

que é justamente um axioma de pré-condição da ação *começar-a-andar*, através do qual pode-se obter a qualificação implícita da referida ação: uma das condições necessárias para a execução de *começar-a-andar* na situação s é a não-validade do fluente *Morto* em s. ■

Resumindo, através de todo esse processo, foi possível determinar os efeitos indiretos das ações (mediante a obtenção do axioma (3.15)), bem como as qualificações implícitas envolvidas (mediante a obtenção do axioma (3.18)), para esse exemplo específico.

Na próxima seção, traçamos uma discussão acerca do formalismo apresentado neste capítulo, realçando seus pontos-chave como abordagem para raciocinar sobre ações.

3.5 Discussão crítica

Uma análise mais atenta da proposta apresentada neste capítulo permite observar que a lógica de base utilizada possui um poder expressivo limitado em relação à sucessão de situações. Ao considerar os axiomas (3.4) e (3.5) na teoria de base, o método proposto em [37] não permite casos como o ilustrado na figura 3.1, a qual representa o fato de se poder obter uma mesma situação a partir de situações e ações diferentes.

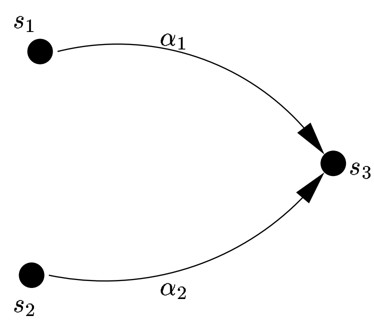

Figura 3.1: Transições entre situações não capturadas pelo cálculo de situações definido em [37]. s_1, s_2 e s_3 representam diferentes situações e α_1 e α_2 ações distintas.

Um exemplo de uma circunstância em que isso pode ocorrer é o próprio cenário do tiro em Yale, apresentado na seção 2.2.2. Suponhamos uma situação em que a vítima esteja viva. Ao se executar a ação *atirar*, na situação resultante ela estará morta. Seja agora outra situação, na qual a vítima já esteja morta. Ao se executar a ação *esperar*, na situação resultante ela também estará morta. É evidente que ambas as situações resultantes são a mesma e fruto de duas situações diferentes mediante a execução de ações diferentes. Casos como esse não são capturados pelo formalismo de base utilizado nessa abordagem.

No tocante aos impactos de a teoria utilizada não ser estratificada, temos que, em situações desse tipo, ficamos impossibilitados de utilizar a compleção de predicados. Nesse sentido, não há outra alternativa senão usar de maneira explícita algum método de circunscrição.

Sendo assim, o fato de ser fundamentada em técnicas de minimização usando circunscrição para tratar teorias não estratificadas compromete a eficiência e a completude deste método, conforme pode ser verificado no trabalho realizado em [66], no qual é feita uma comparação dos principais métodos de circunscrição presentes na literatura.

Outro problema com relação à presente abordagem é a possibilidade de, mesmo tendo-se obtido para cada fluente axiomas de causa da forma de (3.10), não haver como garantir que, para cada fluente, possa-se obter axiomas de estado sucessor da forma de (3.11). Esse eventual problema ocorre devido a ciclos da forma:

$$Holds(Aberta, s) \rightarrow Caused(Aberta, verdadeiro, s), \tag{3.19}$$

e a fórmulas recursivas, como o seguinte axioma de fecho transitivo de domínio:

$$Holds(R(x, z), s) \wedge Holds(R(z, y), s) \rightarrow Caused(R(x, y), verdadeiro, s).$$

Uma possível solução para evitar recursão é usar lógica de segunda ordem, mas isso inviabiliza a implementação de provadores de teoremas para esse formalismo. Quanto à presença de ciclos, segundo palavras do próprio Lin,

there seems to be a sense that these cycles should never arise[5] [37, seção 4].

Uma maneira possível de garantir que esse problema não ocorra, segundo o autor, consiste em evitar escrever axiomas reflexivos como (3.19). Entretanto, esses argumentos são invalidados pelo cenário da porta bloqueada [7], brevemente esboçado a seguir:

Suponhamos uma porta que possa estar fechada ou bloqueada (por uma mesa, por exemplo), e duas ações: abrir, representada a seguir por *abrir*, e arrombar a porta, representada por *arrombar*. A ação *abrir* tem o efeito de abrir a porta, desde que ela não esteja bloqueada, enquanto que *arrombar* abre-a incondicionalmente. Representaremos o fato de a porta estar fechada por *Fechada* e de ela estar bloqueada por *Bloqueada*. A modelagem de tal cenário na abordagem aqui analisada é a seguinte:

$$Caused(Bloqueada, falso, s) \rightarrow Caused(Fechada, falso, s). \tag{3.20}$$

Nesse caso, temos que $Bloqueada \rightarrow Fechada$ (*Fechada* depende de *Bloqueada*). Por outro lado, não é o caso que $Fechada \rightarrow Bloqueada$, uma vez que o axioma

$$Caused(Fechada, falso, s) \rightarrow Caused(Bloqueada, falso, s)$$

não é verdadeiro neste cenário. A explicação é a seguinte: suponha que tenhamos uma

[5] "Parece haver um consenso de que tais ciclos nunca aparecem".

situação na qual o literal *Fechada* é verdadeiro e *Bloqueada* falso. Executando-se a ação *abrir*, temos que *Fechada* é levado a se tornar falso, mas não podemos dizer o mesmo de *Bloqueada*, pois este já era falso antes da ação, e a justificativa para esse raciocínio está no axioma (3.2), que diz explicitamente que *se um fluente é levado a se tornar falso, então ele é falso*, mas não que *se um fluente é falso, então ele foi levado a se tornar falso.*

Com isso, temos que a teoria representada neste cenário é estratificada. Entretanto, conforme apontado em [7], este cenário não é tratado corretamente pelo método de minimização da causalidade sem o enunciado de efeitos indiretos. Sendo assim, estamos diante de uma teoria estratificada para a qual o teorema 3.3.1 não se aplica, constituindo-lhe um contra-exemplo, o que prova a incompletude dessa abordagem.

Antes de finalizarmos esta seção, cabe ainda um último e importante comentário a respeito do enunciado de regras causais, requerido no passo 3 do método aqui apresentado.

Observe que, ao escrevermos fórmulas como (3.20) ou (3.14), estamos, na verdade, enunciando efeitos indiretos de uma maneira explícita. Ou seja, de certa forma esse método requer que sejam determinadas *a priori* as ramificações que se deseja poder deduzir posteriormente. Um exemplo evidente disso é o fato de a fórmula (3.17) ser inferida na teoria às custas de (3.14) (enunciada explicitamente nos passos iniciais).

No decorrer deste trabalho, teremos a oportunidade de retornar à questão de se enunciar efeitos indiretos de maneira explícita na lógica de base.

No próximo capítulo, analisaremos outra abordagem não-monotônica para raciocinar sobre ações baseada em causalidade, que, ao contrário da proposta discutida neste capítulo, não se fundamenta em técnicas de minimização.

Capítulo 4

Derivação *a posteriori* dos efeitos

A primeira condição para se realizar alguma coisa é não querer fazer tudo ao mesmo tempo.

— Tristão de Ataíde

A grande maioria dos formalismos para raciocínio sobre ações presentes na literatura é fundamentada em técnicas de minimização de modelos. Diante das dificuldades apresentadas por esse paradigma face ao problema da ramificação, começaram a surgir propostas que não tinham mais como linha-guia o princípio de minimização.

Uma abordagem causal que acreditamos representar bem essa mudança de enfoque é a proposta por Thielscher [64], a qual baseia-se em um processo de pós-derivação dos efeitos das ações e que teremos a oportunidade de analisar neste capítulo.

A idéia fundamental dessa abordagem consiste em admitir situações que não satisfazem as restrições de domínio, encarando-as como simples estados intermediários. A solução final é obtida depois, mediante a aplicação sucessiva de regras de "correção" no estado inconsistente. Tais regras, chamadas de *relações causais*, constituem a maneira pela qual incorpora-se uma noção de causalidade no formalismo de base e são geradas a partir de uma relação de influência potencial entre fluentes.

4.1 Motivação

Argumentando contra os métodos baseados em minimização, tais quais o de Baker [2], e entre os quais também se encaixa a abordagem apresentada no capítulo anterior, Thielscher afirma que o objetivo de gerar ramificações não consiste em minimizar o conjunto de possíveis mudanças do mundo, mas sim em evitar a ocorrência de mudanças que não sejam causadas, o que realmente faz mais sentido.

Isso é atingido através de uma noção causal explícita expressando quais fatos do domínio são realmente causados e em que condições. Estando essa noção apropriadamente definida, restringem-se os modelos possíveis àqueles nos quais os efeitos indiretos das ações são realmente causados, e não fruto de mudanças espontâneas.

Seguindo esse princípio, com o método de pós-derivação de efeitos considera-se a descrição do mundo obtida após a aplicação de uma dada ação apenas como um resultado intermediário. No estado assim obtido, são determinados os efeitos diretos da ação, sem a necessidade de se preocupar em satisfazer as restrições de domínio. Em seguida, determinam-se os efeitos indiretos através de um processo paralelo de raciocínio. Tal processo se repete até que se tenham exaurido todas as possibilidades de efeitos causados indiretamente e o estado final satisfaça, por fim, as restrições de domínio.

Essa abordagem se assemelha à do capítulo anterior apenas no tocante ao fato de também considerar que os efeitos indiretos são causados não especificamente pela ação propriamente dita, mas pela *presença* de outros efeitos anteriormente gerados.

Isso fica evidente no método de restrição de modelos empregado. Primeiro obtém-se os efeitos diretos da execução de uma ação. Em seguida, informações adicionais são checadas a fim de verificar que outros efeitos devem valer em presença daqueles. Tais efeitos são adicionados e o processo se repete. Em suma, o método de pós-derivação de efeitos também adota uma semântica baseada na idéia fundamental de que efeitos indiretos são gerados ou por efeitos diretos ou por outros efeitos indiretos.

4.2 Lógica de base

Um ponto muito importante da abordagem de pós-derivação de efeitos é a necessidade de se representar o conjunto de todos os fluentes verdadeiros em uma determinada circunstância como um estado. Por esta razão, ao invés de utilizar o cálculo de situações como vínhamos fazendo até então, seguiremos a apresentação original da referida abordagem, na qual é utilizado o cálculo de fluentes [64, 65], que, por ser mais simples que o cálculo de situações, simplificará a apresentação das idéias.

Ao contrário do cálculo de situações, o cálculo de fluentes emprega termos estruturados para representar estados, sendo, portanto, adequado para os propósitos deste método, conforme ficará mais claro na seqüência. Metaforicamente, um estado é uma "cena de um filme", ou seja, uma representação do mundo modelado em um dado instante de tempo[1].

Um *fluente literal* (doravante literal) é um fluente F ou sua negação $\neg F$[2]. Um dado conjunto de literais será inconsistente se contiver, simultaneamente, F e $\neg F$. Como exem-

[1]Apesar da ligeira semelhança dessa interpretação com a do cálculo de situações, neste, entretanto, cada situação é vista como sendo uma seqüência de aplicações sucessivas de ações, estando cada situação, já por definição, "presa" a uma situação anterior.

[2]Lembrando que um fluente é uma propriedade que pode ter seu valor de verdade alterado ao longo do tempo (cf. seção 2.1).

plos de literais, temos, seguindo o clássico exemplo do circuito elétrico de Lifschitz [35], ilustrado na figura 4.1, $\neg Interruptor_1$ e $Interruptor_2$.

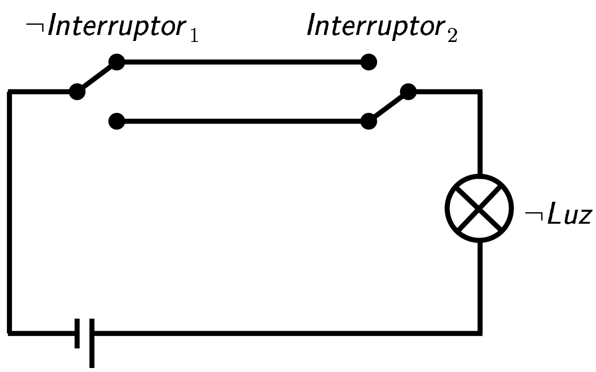

Figura 4.1: O circuito elétrico de Lifschitz [35].

Considerando ainda esse exemplo, temos a seguinte restrição de domínio envolvendo os literais acima e o literal Luz:

$$(Interruptor_1 \leftrightarrow Interruptor_2) \leftrightarrow Luz. \tag{4.1}$$

que denota que "sempre que os dois interruptores estiverem na mesma posição, então a luz se acende".

Um *estado* é um conjunto maximal consistente de literais. Exemplo de um estado possível seria

$$S = \{\neg Interruptor_1, Interruptor_2, \neg Luz\}.$$

Note que, nesse caso, S satisfaz a restrição de domínio (4.1).

Se L é um literal, denotamos por $|L|$ o seu componente positivo (átomo), ou seja, $|F| = |\neg F| = F$, com F um fluente. Como exemplo,

$$|\neg Interruptor_1| = Interruptor_1.$$

Tal notação se estende também a conjuntos de literais S:

$$|S| = \{|L| : L \in S\}.$$

Todo literal é uma fórmula. $\top$ (tautologia) e $\perp$ (contradição) são fórmulas. Se A e B são fórmulas, também o é $A \wedge B$. Utilizam-se $A \vee B$, $A \rightarrow B$ e $A \leftrightarrow B$ como abreviaturas no sentido usual.

A noção de verdade (respectivamente falsidade) de um literal L em relação a um dado estado S é definida como L sendo verdadeiro (respectivamente falso) em S se e somente se $L \in S$ (respectivamente $L \notin S$). Assim, a semântica das fórmulas em um dado estado S é dada pela seguinte definição:

Definição 4.2.1 Sejam A e B fórmulas e S um estado.

- $\top$ é verdadeira e $\bot$ falsa em S;
- Um literal L é verdadeiro em S se e somente se $L \in S$;
- $A \wedge B$ é verdadeira em S se e somente se A e B forem verdadeiras em S; ■

Da mesma forma, definem-se as condições de verdade de $A \vee B$, $A \rightarrow B$ e $A \leftrightarrow B$ da maneira usual.

Uma *lei de ação* é uma tripla $\langle C, \alpha, E\rangle$, onde C e E são conjuntos consistentes[3] de literais tais que $|C| = |E|$, e α é o nome de uma ação.

O conjunto C é chamado de *condição* da lei de ação e constitui o conjunto de literais presentes no estado atual que devem valer para que a lei de ação em questão possa ser aplicada. E representa o conjunto de *efeitos diretos* da execução da ação α e é constituído pelos literais que devem valer no estado resultante da aplicação da referida lei de ação.

Um exemplo de lei de ação é

$$\langle\{\neg \mathit{Interruptor}_1\}, \mathit{apertar}_1, \{\mathit{Interruptor}_1\}\rangle, \tag{4.2}$$

que denota o fato de que, dada uma condição na qual $\neg \mathit{Interruptor}_1$ valha, após executar a ação $\mathit{apertar}_1$, que significa alterar a posição de $\mathit{Interruptor}_1$, tem-se como resultado que $\mathit{Interruptor}_1$ vale.

Uma lei de ação $a = \langle C, \alpha, E\rangle$ é *aplicável* a um estado S se e somente se $C \subseteq S$. Como exemplo, a lei de ação (4.2) é aplicável ao estado

$$\{\neg \mathit{Interruptor}_1, \mathit{Interruptor}_2, \neg \mathit{Luz}\}.$$

Note que a aplicação de a a S resulta no conjunto $(S \setminus C) \cup E$.

Devido ao fato de S ser um estado (e portanto consistente), C e E dois conjuntos consistentes e $|C| = |E|$, tem-se que o resultado da aplicação de toda lei de ação resulta também num estado.

Entretanto, é importante ressaltar que o novo estado assim obtido não necessariamente satisfaz as restrições de domínio, uma vez que apenas os efeitos diretos das ações estão

[3]Consistentes no sentido usual.

especificados. Para o exemplo anterior, temos que o estado resultante da aplicação da lei de ação considerada é

$$\{Interruptor_1, Interruptor_2, \neg Luz\},$$

o qual evidentemente não satisfaz a restrição de domínio (4.1).

Na próxima seção, discorreremos sobre a maneira pela qual a abordagem aqui apresentada resolve as incoerências de tais estados intermediários na busca pela solução desejada.

4.3 Método empregado

Nesta seção, apresentamos uma maneira de se calcular todos os efeitos indiretos das ações, obtendo um estado consistente ao final do processo.

Conforme ilustrado no final da seção precedente, há a possibilidade de o estado resultante da aplicação de uma lei de ação contradizer as restrições de domínio. Neste caso, utiliza-se uma noção causal direcionada envolvendo literais específicos de maneira a "corrigir" as inconsistências do estado obtido.

Definição 4.3.1 Uma *relação causal* é uma expressão da forma L_1 causes L_2 if Φ, na qual Φ é uma fórmula e L_1 e L_2 são literais. ∎

Um exemplo de relação causal é

$$Interruptor_1 \text{ causes } Luz \text{ if } Interruptor_2. \tag{4.3}$$

Uma relação causal L_1 causes L_2 if Φ é *aplicável* a um par (S, E), constituído de um estado S e de um conjunto consistente E, se e somente se $\Phi \wedge \neg L_2$ for verdadeira em S e $L_1 \in E$.

Como resultado de tal aplicação, tem-se o par (S', E'), no qual $S' = (S \setminus \{\neg L_2\}) \cup \{L_2\}$ e $E' = (E \setminus \{\neg L_2\}) \cup \{L_2\}$. Exemplificando, a relação causal (4.3) acima é aplicável ao par

$$(\{Interruptor_1, Interruptor_2, \neg Luz\}, \{Interruptor_1\}),$$

o que dará como resultado o par

$$(\{Interruptor_1, Interruptor_2, Luz\}, \{Interruptor_1, Luz\}).$$

Seja $\mathcal{R}$ um conjunto de relações causais definidas, então $(S, E) \longmapsto_{\mathcal{R}} (S', E')$ denota a existência de um elemento de $\mathcal{R}$ cuja aplicação a (S, E) resulta em (S', E').

Tudo o que acima foi definido quer dizer que uma relação causal é aplicável se a condição associada Φ for verdadeira no estado em questão e o possível efeito indireto L_2

for falso[4], além de L_1 estar entre os efeitos anteriormente gerados, sejam eles diretos ou indiretos. Como S é um estado e E é consistente, $(S, E) \longmapsto_{\mathcal{R}} (S', E')$ implica S' ser um estado e E' consistente também.

Uma seqüência de relações causais $r_1, \ldots, r_n$ é *aplicável* ao par (S_0, E_0) se e somente se existirem pares $(S_1, E_1), \ldots, (S_n, E_n)$, tais que, para cada $1 \leq i \leq n$, r_i é aplicável a (S_{i-1}, E_{i-1}), resultando em (S_i, E_i). Nesse caso, utiliza-se $(S, E) \overset{*}{\longmapsto}_{\mathcal{R}} (S', E')$ para denotar a existência de uma seqüência (possivelmente vazia) de relações causais em $\mathcal{R}$ que seja aplicável a (S, E) e resulte em (S', E').

Suponhamos que esteja definido um conjunto de relações causais e que após a aplicação de uma lei de ação obteve-se um conjunto de fluentes S, contendo os efeitos diretos da ação em questão, e que S viole alguma restrição de domínio. Nesse caso, no sentido de determinar um estado resultante que seja satisfatório, calculam-se os efeitos indiretos através da seleção (a princípio não-determinística) e aplicação serial de relações causais. Se este procedimento, eventualmente, resultar num estado que satisfaz as restrições de domínio, obtém-se então um *estado sucessor*.

Em síntese, o processo todo é resumido pela seguinte definição:

Definição 4.3.2 Sejam $\mathcal{D}$ um conjunto de restrições de domínio, $\mathcal{R}$ um conjunto de relações causais, S um estado que satisfaz $\mathcal{D}$, α uma ação e C e E conjuntos consistentes tais que $|C| = |E|$. Um estado S' é um *estado sucessor* de S e α se e somente se houver uma lei de ação $\langle C, \alpha, E \rangle$ aplicável a S tal que:

1. $(S, E) \overset{*}{\longmapsto}_{\mathcal{R}} (S', E')$, para algum E', e
2. S' satisfaz $\mathcal{D}$. ■

A manutenção do elemento E de cada par (S, E) é uma forma de saber de que estado se está vindo, ou seja, a abordagem de pós-derivação de efeitos permite que de dois estados diferentes se obtenha um mesmo estado resultante, não estando, portanto, sujeita à limitação do formalismo de base do método de minimização da causalidade apresentada na figura 3.1.

Vamos nos ater agora ao processo de obtenção das relações causais. Como é de se esperar, as mesmas são calculadas tomando-se por base as restrições de domínio, uma vez que o estado final resultante de sua aplicação deverá satisfazê-las. Porém, uma avaliação

[4] $\Phi \wedge \neg L_2$ pode ser interpretada como o *contexto* no qual uma relação causal é aplicada. Entretanto, em [64] não é adotada essa interpretação. Voltaremos a considerar a noção de contexto conjuntamente ao de causalidade no decorrer do presente trabalho.

puramente sintática das restrições de domínio pode levar à obtenção de relações causais totalmente contra-intuitivas.

Como exemplo, seja a restrição (4.1). Percebe-se que tal fórmula acarreta

$$(Interruptor_1 \wedge \neg Luz) \rightarrow \neg Interruptor_2,$$

que, por sua vez, acarreta a fórmula

$$Interruptor_1 \rightarrow (\neg Luz \rightarrow \neg Interruptor_2),$$

a partir da qual se obteria a relação causal

$$Interruptor_1 \textsf{ causes } \neg Interruptor_2 \textsf{ if } \neg Luz,$$

o que é completamente contra o esperado.

Sendo assim, temos que as restrições de domínio não fornecem informação suficiente para a determinação das relações causais.

Para auxiliar a obtenção das relações causais evitando resultados não-intuitivos, define-se uma relação indicando quais fluentes *podem* influenciar o valor de verdade de outros:

Definição 4.3.3 Seja $\mathcal{F}$ um conjunto de fluentes. Uma *relação de influência* $\mathcal{I}$ é uma relação binária sobre $\mathcal{F} \times \mathcal{F}$. ■

Com isso, $(F, F') \in \mathcal{I}$ representa a informação de que uma mudança no valor de verdade de F *pode acarretar* uma mudança no de F'. Nesse sentido, o método aqui apresentado difere substancialmente do analisado no capítulo anterior na medida em que apenas *permite* mudança, enquanto que aquele realmente *causa* mudança.

Uma instância de relação de influência para o exemplo considerado até então seria

$$\mathcal{I} = \{(Interruptor_1, Luz), (Interruptor_2, Luz)\},$$

que representa exatamente o fato intuitivo de que uma alteração em qualquer dos dois interruptores *pode* acarretar uma alteração na lâmpada.

As restrições de domínio e a relação de influência provêem informação suficiente para a geração automática de um conjunto adequado de relações causais. A idéia básica consiste em investigar todas as possibilidades de violação de uma restrição de domínio, codificando simultaneamente as relações causais que possam "corrigi-las". Tal idéia é formalizada através do seguinte algoritmo:

Algoritmo 4.3.1 (Geração de relações causais)

entrada: A relação de influência $\mathcal{I}$;

A forma normal conjuntiva do conjunto $\mathcal{D}$ de restrições de domínio, $CNF(\mathcal{D})$;

saída: O conjunto $\mathcal{R}$ de relações causais;

$\mathcal{R} \leftarrow \{\}$;

Para cada $D_i = L_1 \vee \cdots \vee L_{m_i} \in CNF(\mathcal{D})$, $i = 1, \ldots, n$

Para cada $j = 1, \ldots, m_i$

Para cada $k = 1, \ldots, m_i$, $k \neq j$

Se $(|L_j|, |L_k|) \in \mathcal{I}$, adicionar a $\mathcal{R}$ a relação causal:

$$\neg L_j \text{ causes } L_k \text{ if } \bigwedge_{\substack{l=1,\ldots,m_i \\ l\neq j, l\neq k}} \neg L_l$$

■

Uma maneira de implementar essa solução é através de programação lógica [12], tal qual realizado no trabalho original em [64]. No referido artigo, explora-se a lógica de predicados de primeira ordem através da definição de um cálculo de fluentes com unificação.

Na próxima seção, apresentaremos alguns cenários-exemplos de aplicação do método de pós-derivação de efeitos, a fim de melhor verificar suas principais características.

4.4 Cenários-exemplos

Apresentamos na seqüência dois cenários-exemplos empregando o método até então discutido. No primeiro deles, o objetivo consiste em poder derivar tanto os efeitos indiretos como as qualificações implícitas [23, 40] para a execução das ações. No segundo, o objetivo será testar a aplicação da abordagem de pós-derivação de efeitos a um caso envolvendo uma transição não-determinística.

Exemplo 4.4.1 (A caça ao peru [65]) Este cenário-exemplo consiste de uma adaptação dos cenários do tiro em Yale [27] e do peru que caminha [2]. Neste caso, deseja-se caçar um peru, o qual pode estar vivo, fato este representado pelo literal *Vivo*, ou morto, representado pelo literal $\neg$*Vivo*. O peru também pode estar caminhando, representado por *Caminhando*, ou parado, $\neg$*Caminhando*. Obviamente, temos também uma restrição de domínio especificando que um peru que esteja caminhando obrigatoriamente está vivo:

$$Caminhando \rightarrow Vivo. \tag{4.4}$$

Completando a teoria, duas ações são possíveis: atirar no peru, representada por *atirar*, cujo efeito é justamente matar o animal, e atrair o peru (jogando um punhado de milho, por exemplo), representada por *atrair*, cujo efeito é fazer com que o peru comece a caminhar. Em função disso, define-se, para este cenário, duas leis de ações possíveis:

$$\langle\{\mathit{Vivo}\}, \mathit{atirar}, \{\neg\mathit{Vivo}\}\rangle, \tag{4.5}$$

$$\langle\{\neg\mathit{Caminhando}\}, \mathit{atrair}, \{\mathit{Caminhando}\}\rangle \tag{4.6}$$

Seguindo as definições da seção anterior, estabelece-se para este exemplo a relação de influência como sendo

$$\mathcal{I} = \{(\mathit{Vivo}, \mathit{Caminhando})\},$$

o que representa o fato de que uma mudança em *Vivo* pode acarretar uma mudança em *Caminhando*.

Utilizando a restrição de domínio (4.4) e a relação de influência acima e aplicando o algoritmo 4.3.1, obtém-se a seguinte relação causal:

$$\neg\mathit{Vivo} \text{ causes } \neg\mathit{Caminhando} \text{ if } \top \tag{4.7}$$

o que está de acordo com a intuição ao expressar que o fato de *Vivo* deixar de valer faz com que imediatamente *Caminhando* deixe de valer também, em qualquer circunstância.

Considerando, agora, o par inicial

$$(S_0, E_0) = (\{\mathit{Vivo}, \neg\mathit{Caminhando}\}, \{\}),$$

o qual contém o estado inicial $\{\mathit{Vivo}, \neg\mathit{Caminhando}\}$, ao se aplicar a lei de ação (4.5), basta que se determine o efeito direto $\neg\mathit{Vivo}$, obtendo-se, assim, o par

$$(S_1, E_1) = (\{\neg\mathit{Vivo}, \neg\mathit{Caminhando}\}, \{\neg\mathit{Vivo}\}).$$

Já se o estado inicial for $\{\mathit{Vivo}, \mathit{Caminhando}\}$, a execução da ação *atirar* fará com que o peru morra e, de acordo com a restrição de domínio (4.4) acima, deveremos concluir que o animal já não estará mais caminhando. Isso é obtido mediante a aplicação da relação causal (4.7) acima ao estado intermediário $\{\neg\mathit{Vivo}, \mathit{Caminhando}\}$. Tal relação captura o fato de, sendo $\neg\mathit{Vivo}$ um efeito direto produzido, sua presença *causa* a de $\neg\mathit{Caminhando}$, resultando no estado final $\{\neg\mathit{Vivo}, \neg\mathit{Caminhando}\}$, conforme desejado.

Da mesma forma, se considerarmos a aplicação da lei de ação (4.6) ao estado inicial $\{\mathit{Vivo}, \neg\mathit{Caminhando}\}$, será suficiente obter o efeito direto *Caminhando*. Por outro lado, se se aplicar a mesma lei de ação ao par

$$(S'_0, E'_0) = (\{\neg Vivo, \neg Caminhando\}, \{\}),$$

o resultado será completamente diferente. Nesse caso, obter-se-á o par intermediário

$$(S'_1, E'_1) = (\{\neg Vivo, Caminhando\}, \{Caminhando\}),$$

o qual também viola a restrição de domínio.

No entanto, a relação (4.7) não é aplicável ao par (S'_1, E'_1), pois $\neg Vivo$ não foi causado como um efeito. Esta sutileza é capturada através da consulta ao elemento E'_1 do par (S'_1, E'_1), ao qual a relação causal é aplicada: como $\neg Vivo \notin E'_1$, conclui-se que $\neg Vivo$ não é um efeito causado. Assim, pela definição 4.3.2, conclui-se que não existe um estado sucessor, o que determina uma qualificação implícita (cf. seção 3.4) da ação *atrair*. ■

Nesse sentido, o formalismo de pós-derivação dos efeitos das ações permite, para esse exemplo, a obtenção do efeito indireto da ação *atirar* bem como a determinação de uma qualificação implícita para a ação *atrair*.

Toda vez em que não for possível determinar um estado sucessor, significa que foi capturada uma qualificação implícita da ação em consideração.

Por outro lado, a ocorrência do caso oposto, isto é, a presença de mais de um estado sucessor, caracteriza uma ação não-determinística. O exemplo a seguir ilustra uma instância deste último caso.

Exemplo 4.4.2 (Acender uma lâmpada [60]) Considere um cenário com uma lâmpada (Luz) conectada a dois interruptores ($Interruptor_1$ e $Interruptor_2$) em paralelo. A restrição de domínio, para este caso, é dada por

$$Luz \leftrightarrow Interruptor_1 \vee Interruptor_2. \tag{4.8}$$

A idéia por detrás deste cenário consiste em introduzir uma nova ação, *acender* (em adição a $apertar_1$ e $apertar_2$), que representa o fato de se acender diretamente a lâmpada. Nesse caso, o efeito direto observável é de que a lâmpada se acende, enquanto que as alterações dos interruptores são vistas como efeitos indiretos.

Sendo assim, modelando este cenário, tem-se a lei de ação

$$\langle\{\neg Luz\}, acender, \{Luz\}\rangle, \tag{4.9}$$

e a relação de influência dada por

$$\mathcal{I} = \{(Luz, Interruptor_1), (Luz, Interruptor_2)\}.$$

A partir dessas informações, pode-se obter as seguintes relações causais:

$$Luz \textsf{ causes } Interruptor_1 \textsf{ if } \top \tag{4.10}$$

$$Luz \textsf{ causes } Interruptor_2 \textsf{ if } \top \tag{4.11}$$

Supondo a situação inicial

$$(S_0, E_0) = (\{\neg Interruptor_1, \neg Interruptor_2, \neg Luz\}, \{\}),$$

após a aplicação da lei de ação (4.9), obtém-se

$$(S_0', E_0') = (\{\neg Interruptor_1, \neg Interruptor_2, Luz\}, \{Luz\}), \tag{4.12}$$

que não satisfaz a restrição de domínio (4.8) acima. Como a relação causal (4.10) é aplicável a esse par, temos

$$(S_1, E_1) = (\{Interruptor_1, \neg Interruptor_2, Luz\}, \{Luz, Interruptor_1\}), \tag{4.13}$$

o qual, por sua vez, satisfaz a restrição de domínio (4.8), não havendo, portanto, a necessidade de uma nova aplicação de uma relação causal. Entretanto, observando o par (4.12), percebemos que é possível também lhe aplicar a relação causal (4.11), o que resultará em

$$(S_2, E_2) = (\{\neg Interruptor_1, Interruptor_2, Luz\}, \{Luz, Interruptor_2\}),$$

como resultado intermediário. Da mesma forma, o estado deste último par satisfaz a restrição de domínio (4.8) e, tal qual (4.13), também constitui um estado sucessor de (4.12).

Entretanto, observe que é possível ainda um terceiro estado sucessor,

$$(S_3, E_3) = (\{Interruptor_1, Interruptor_2, Luz\}, \{Luz, Interruptor_1, Interruptor_2\}),$$

interpretado como resultado da alteração dos dois interruptores ao mesmo tempo e que não é obtido com a abordagem aqui apresentada. Por outro lado, com a extensão do método de minimização da causalidade apresentado em [38], tal estado pode ser capturado. ■

Na próxima seção, teceremos alguns comentários sobre o método aqui analisado, ilustrando seus pontos positivos e as principais falhas descobertas em estudos subseqüentes.

4.5 Discussão crítica

Um ponto positivo da abordagem analisada neste capítulo, segundo apontado em [65], é o fato de possibilitar a geração automática de todas as relações causais adequadas para um dado domínio. Segundo seu autor, isso constitui uma grande vantagem sobre outros formalismos, dentre os quais o de minimização da causalidade, que deixam a cargo do projetista a construção das leis que definem a noção causal.

Ora, mas se, por um lado, o método aqui discutido é capaz de obter todas as relações causais a partir das restrições de domínio e da relação de influência, por outro a obtenção desta última não se faz senão também por meio de um método supervisionado. Assim, de nada adianta exonerar o projetista da tarefa de estabelecer as relações causais se ainda assim ele tiver de empreender tarefa semelhante para estabelecer as influências potenciais dos fluentes.

Thielscher [65] afirma que todo estado sucessor obtido através de algum método de minimização também pode ser obtido mediante o uso de sua abordagem. Esse argumento, entretanto, é invalidado pelo exemplo 4.4.2, no qual vimos que existe um estado possível que não pode ser capturado pelo seu método.

No tocante ao formalismo de base, para que seja possível explorar todo o poder expressivo da lógica de primeira ordem, faz-se necessária uma redefinição das relações causais para que comportem também variáveis[5].

Entretanto, essa extensão acarreta alguns problemas. Como exemplo, a geração automática das relações causais fica mais complicada, uma vez que não está bem claro o relacionamento entre as possíveis instâncias de fórmulas com variáveis e o processo de "correção" das circunstâncias violadoras das restrições de domínio.

No que se refere à implementação da abordagem aqui analisada, em [65] é esboçada sua codificação em programação lógica através da definição do predicado *Successor*. Tal predicado é definido em termos de outro predicado, *Ramify*, que tem por objetivo estabelecer o fecho transitivo do espaço de estados possíveis, o que é usado para que se possa determinar os conjuntos S' e E' da definição 4.3.2. Infelizmente, a única maneira de se proceder quanto a isso é utilizando uma fórmula em lógica de segunda ordem, o que compromete a completude e a decidibilidade da solução para domínios mais complexos.

Uma vantagem da pós-derivação de efeitos em relação à minimização da causalidade consiste em sua maior robustez diante de domínios envolvendo "dependências cíclicas", tais como as discutidas na seção 3.5. Além do mais, conforme apontado em [65], a ocorrência de relações de influência cíclicas não acarreta necessariamente seqüências infinitas de aplicações sucessivas de relações causais.

[5] O método apresentado na seção 4.3 do presente trabalho considera uma versão mais simples do cálculo de fluentes, justamente para facilitar sua compreensão. Para que a abordagem de pós-derivação de efeitos seja realmente útil, requer-se uma extensão, tal qual apresentada em [65, seção 6].

Em termos de completude do método apresentado neste capítulo, temos que há de fato casos em que sua utilização não produz os resultados desejados.

Conforme demonstrado em [7], para o cenário da porta bloqueada, brevemente definido na seção 3.5, a utilização da presente abordagem faz com que o par $(Fechada, Bloqueada)$, em função da ação *arrombar*, pertença à relação de influência. Isso faz com que seja impossível derivar a qualificação implícita para a ação *abrir*.

Antes de finalizarmos este capítulo, apresentamos ainda outro exemplo para o qual a aplicação da pós-derivação dos efeitos de ações não é correta.

Exemplo 4.5.1 (A maleta de Toulouse [7]) Considere uma maleta cujos dois prendedores estejam ligados (por uma barra de metal, por exemplo) e sempre na mesma posição, fato este representado pelo literal $Conectados$. O fato de o primeiro (respectivamente segundo) prendedor estar erguido é representado por $ParaCima_1$ (respectivamente $ParaCima_2$). O fato de a maleta estar aberta é dado por $Aberta$. Para este cenário, temos as seguintes restrições de domínio:

$$Conectados \rightarrow (ParaCima_1 \leftrightarrow ParaCima_2)$$

$$(ParaCima_1 \wedge ParaCima_2) \rightarrow Aberta$$

A relação de influência correspondente é dada por:

$$\mathcal{I} = \left\{ \begin{array}{c} (ParaCima_1, Aberta), \\ (ParaCima_2, Aberta), \\ (ParaCima_1, ParaCima_2), \\ (ParaCima_2, ParaCima_1) \end{array} \right\}$$

o que leva à geração das relações causais incorretas:

$$ParaCima_1 \textsf{ causes } \neg ParaCima_2 \textsf{ if } \neg Aberta$$

e

$$ParaCima_2 \textsf{ causes } \neg ParaCima_1 \textsf{ if } \neg Aberta$$

Esse tipo de problema ocorre justamente com abordagens que apresentam uma noção causal indexada por literais, que é o caso dos métodos analisados nestes dois últimos capítulos, o que lança as bases para uma investigação maior sobre causalidade indexada por ações [7]. Retornaremos ainda a esse tópico no decorrer do presente trabalho. ■

A partir do próximo capítulo, passaremos a analisar as abordagens ditas monotônicas para raciocinar sobre ações.

Capítulo 5

Fecho de explicação

A falta de alternativas clarifica maravilhosamente a mente.

— Henry Kissinger

Neste capítulo, analisaremos a abordagem apresentada por Schubert [61], a qual influenciou de maneira decisiva muitas das pesquisas subseqüentes na área.

Ao contrário das propostas analisadas nos capítulos precedentes, esta consiste de um formalismo monotônico para raciocinar sobre ações e apresenta uma noção de causalidade implícita para tratar os problemas da persistência e da ramificação.

Nesta abordagem, propõe-se a codificação no cálculo de situações de todas as condições necessárias para a ocorrência de mudanças no mundo como uma alternativa aos axiomas de persistência. Com isso, obtém-se uma maneira simples para resolver, de forma monotônica, a grande maioria dos problemas para os quais se acreditava, até então, serem possíveis apenas soluções baseadas em raciocínio não-monotônico.

5.1 Fundamentos

A abordagem de fecho de explicação consiste em um formalismo monotônico para a realização de descrições de domínio. Dada uma teoria representando um cenário no cálculo de situações, por exemplo, a idéia fundamental consiste na definição de um conjunto de *axiomas de fecho de explicação*[1] [61], combinados com a hipótese de fecho de ações[2] [62].

Axiomas de fecho de explicação constituem uma maneira formal de se especificar *condições necessárias* à alteração efetiva de uma determinada propriedade. Tais condições são representadas pela execução de ações e é neste ponto que fica evidente sua noção causal, uma vez que mudanças ocorrem somente através da execução de ações, sendo

[1] No original em inglês, *explanation closure – EC – axioms.*

[2] No original em inglês, *action closure – AC.*

causadas por estas. O esquema básico de um axioma de fecho de explicação é dado por:

$$\begin{array}{lcl} \textit{mudança observada} & \rightarrow & \textit{ação executada, de uma certa forma,} \\ & & \textit{em um determinado contexto.} \end{array}$$

Tal esquema é lido como: um dado efeito é produzido *somente se* a ação que o produz for executada de tal maneira e em tais condições.

Com isso, os axiomas de fecho de explicação, além de constituírem uma maneira sucinta de representação, permitem uma forma adequada de inferir as não-mudanças do mundo, sendo, portanto, uma alternativa viável aos axiomas de persistência, conforme ficará mais claro a seguir.

A idéia original por detrás dos axiomas de fecho de explicação foi proposta por Haas [26], que denominou tais axiomas de *axiomas de persistência específicos de domínio*[3]. Schubert [61] aproveitou a idéia de Haas, modificando sutilmente a semântica de base. Ao invés de enxergar tais axiomas sob o prisma de axiomas de persistência, propôs uma visão segundo a qual esse tipo de axioma fornece, por definição, o conjunto completo de todas as explicações possíveis para uma dada mudança ocorrer[4].

Uma vez definido um conjunto adequado de axiomas de fecho de explicação, e como as leis de efeito das ações permitem a representação do que realmente muda após executar uma ação, combinando estas com aqueles, obtém-se um formalismo capaz de representar tanto as mudanças quanto as não-mudanças em um mundo dinâmico. Respaldando essa idéia está a chamada hipótese de fecho de ações, a qual especifica que *todas* as ações envolvidas na representação de uma dada *parcela* do domínio em especial são conhecidas.

Com isso, obtém-se um formalismo correto, implementável e eficiente, que constitui uma alternativa viável aos métodos não-monotônicos e baseados em circunscrição.

A seguir, apresentamos a semântica e a axiomática definidas para o formalismo do fecho de explicação.

5.2 Semântica de base e axiomática

A abordagem de fecho de explicação utiliza a lógica do cálculo de situações, tal qual apresentado na seção 2.1.

Em termos de semântica, a grande maioria dos formalismos presentes na literatura possui como idéia central a visão de um mundo completamente inerte, no qual todos os

[3]No original em inglês, *domain-specific frame axioms*.

[4]De certa maneira, os axiomas de fecho de explicação podem ser enxergados como *condições necessárias mais fortes* [39].

fluentes envolvidos mantêm seus respectivos valores de verdade *exceto* quando são forçados a alterá-los em função de ações executadas pelos agentes.

Por discordar desse princípio, o autor desta abordagem define seu formalismo de uma maneira completamente desprovida da semântica inercial, argumentando que o mundo, por mais que *pareça* inerte, na realidade *não o é*. A justificativa para tal reside no fato de que mudanças ocorrem a todo instante, desde o micro até o macrocosmo, e que nesse sentido a hipótese da inércia, pelo menos tal como entendida na comunidade, simplesmente não condiz com a realidade [62].

Com isso, a semântica de base da abordagem de fecho de explicação não procura focalizar as não-mudanças do mundo, mas sim as condições consideradas razoáveis para que determinadas mudanças ocorram. As não-mudanças, nesse sentido, são derivadas indiretamente a partir da não-verificabilidade de tais condições.

No tocante à axiomática básica definida para esse formalismo, temos que o único axioma universal, isto é, não-dependente de domínio, é o que estabelece que ações com nomes diferentes são consideradas como sendo distintas. Isso significa que, seguindo as definições da seção 2.1,

$$\forall \alpha, \beta \in \mathsf{AÇÕES}(\alpha \textit{ e } \beta \textit{ são nomes diferentes} \rightarrow \alpha \neq \beta). \tag{5.1}$$

Esse axioma é fundamental no processo de determinação da persistência usando axiomas de fecho de explicação, conforme será ilustrado na seqüência.

Além do axioma (5.1), considera-se ainda a hipótese de fecho de ações, a qual está implícita no próprio cálculo de situações, fundamentada na dependência funcional existente entre situações e ações.

Na seqüência, ilustramos como que os axiomas de fecho de explicação codificados no cálculo de situações são utilizados no estabelecimento de um formalismo para se raciocinar em mundos dinâmicos.

5.3 Método empregado

Uma vez definida uma lógica de base e estando enunciado o axioma (5.1), o método aqui apresentado consiste pura e simplesmente na codificação de axiomas de fecho de explicação apropriados na lógica em questão.

Como exemplo, suponhamos um cenário envolvendo um robô, representado por R, que pode estar segurando um objeto x numa dada situação s, fato este representado por $Holds(Segurando(R, x), s)$, e que pode executar a ação de soltar o referido objeto, representada por $soltar$. Para este cenário, codificaríamos o seguinte axioma de fecho de explicação (supondo x e s universalmente quantificadas):

$$(Holds(Segurando(R,x),s) \wedge \neg Holds(Segurando(R,x), result(\alpha,s)) \rightarrow \alpha = soltar), \qquad (5.2)$$

que significa que "se o robô estava segurando um certo objeto e depois de executar uma certa ação não o está mais, então necessariamente tal ação tem de ser a de soltar o referido objeto". Observe como o axioma (5.2) encerra em si a explicação para uma mudança no fato de o robô estar segurando o objeto em questão.

Suponhamos, agora, que o robô pode ainda realizar outras ações, como, por exemplo, a de pintar o objeto que esteja segurando, representada por *pintar*.

Sendo assim, supondo que numa determinada situação o robô esteja segurando um objeto específico B e que na mesma situação ele executa a ação de pintar o referido objeto, pode-se facilmente inferir pela contrapositiva de (5.2) que, na situação resultante de executar a ação *pintar*, o robô continua ainda segurando o objeto B.

Note o papel sutil, porém fundamental, representado pelo axioma (5.1) durante todo esse processo: como a ação executada na situação s foi *pintar*, e como o valor de $Holds(Segurando(R,x),s)$ só mudará na ocorrência, em s, da ação *soltar*, o fato de, por (5.1), termos $pintar \neq soltar$ garante a permanência de $Segurando(R,x)$.

Portanto, não há necessidade de se enunciar axiomas de persistência relatando que a propriedade $Segurando(R,x)$ permanece inalterada após a ação *pintar*. Essa tarefa é realizada pelos axiomas de fecho de explicação definidos para o domínio em conjunção com o axioma (5.1). É exatamente dessa maneira que a presente abordagem resolve o problema da persistência.

Através desse exemplo, fica também mais clara a noção de causalidade implícita traduzida pelos axiomas de fecho de explicação: as mudanças são *causadas* somente se as ações que as produzem forem realmente executadas. Em outros termos, não há mudanças espontâneas.

Observe que, além da elaboração dos axiomas de fecho de explicação, é importante também para essa abordagem o papel desempenhado pela hipótese de fecho de ações, que, conforme anteriormente citado, tem por objetivo estabelecer que não há ações "misteriosas" ou "desconhecidas" para a fatia do domínio em representação.

A importância dessa hipótese pode ser vista no próprio exemplo acima. Se não houvesse a hipótese de que as únicas ações possíveis são *soltar* e *pintar*, nada nos impediria de supor uma ação *teletransportar* sobre o objeto B que também produziria o efeito de o robô não o estar mais segurando. Se esse fosse o caso, nenhuma conclusão efetiva desse cenário tal como descrito poderia ser obtida.

Retornaremos à questão da hipótese de fecho de ações na discussão crítica ao final deste capítulo.

Estando, assim, a teoria que representa o domínio devidamente codificada e contendo os axiomas de fecho de explicação adequados, emprega-se uma implementação baseada em STRIPS [16] para simular o processo de raciocínio. Basicamente, o ponto-chave da implementação esboçada em [61] reside no fato de os axiomas de fecho de explicação poderem ser diretamente traduzidos em eficientes métodos de inferência em STRIPS.

Para implementar a abordagem aqui analisada, é empregada a "estratégia do cão dorminhoco"[5]: propriedades tornadas verdadeiras ou falsas ao longo do processo são atualizadas, sendo que o resto supõe-se que permanece inalterado.

A idéia básica consiste em produzir os efeitos dos passos de cada plano realizado inferindo a persistência mediante regras *default*. Estas realizam a tarefa de considerar os valores mais recentes dos fluentes como ainda válidos na situação em questão. O papel desempenhado pelos axiomas de fecho de explicação, nesse caso, é o de evitar que as regras *default* acabem produzindo persistências muito fortes e indesejadas, principalmente quando se consideram descrições incompletas de domínio. Com isso, garante-se que as inferências assim realizadas são corretas com relação a uma teoria contendo os axiomas de fecho de explicação adequados.

Em linhas gerais, a proposta de fecho de explicação é bastante simples, pois reúne idéias anteriormente debatidas na literatura sob o prisma do raciocínio monotônico, empregando uma semântica simples e baseada num método de planificação bastante conhecido. Além do mais, tal abordagem se mostra correta para muitos dos cenários estudados na área.

A seguir, têm-se alguns cenários-exemplos que bem ilustram a modelagem de alguns problemas segundo a proposta aqui apresentada e que servirão para explicitar os detalhes acerca do método empregado.

5.4 Cenários-exemplos

Em [62, seção 2], o autor submete sua abordagem[6] à bateria de testes sugerida em [59] e mostra que muitas das tarefas em raciocínio sobre ações que se pensava requererem métodos não-monotônicos podem de fato ser resolvidas monotonicamente.

O objetivo desta seção, porém, é ilustrar a aplicação da abordagem de fecho de expli-

[5] No original em inglês, *sleeping dog strategy*.

[6] Entretanto, no referido artigo, é utilizado o formalismo de *Features and Fluents* [59] ao invés do cálculo de situações, justamente com o intuito de mostrar que os axiomas de fecho de explicação independem da lógica de base utilizada.

cação a apenas dois cenários, sendo um o cenário do tiro em Yale [27], que bem ilustra o problema da persistência, e outro o cenário da caça ao peru [65], que é um exemplo envolvendo ramificação. Para mais exemplos de aplicação do método aqui analisado, remetemos ao trabalho acima referido.

Exemplo 5.4.1 (O cenário do tiro em Yale [27]) Para o cenário do tiro em Yale, apresentado na seção 2.2.2, a modelagem usando fecho de explicação é empreendida da seguinte maneira:

$$Holds(Carregada, result(carregar, s)) \tag{5.3}$$

$$Holds(Carregada, s) \rightarrow \neg Holds(Vivo, result(atirar, s)) \tag{5.4}$$

$$Holds(Vivo, s_0) \tag{5.5}$$

$$\neg Holds(Carregada, s_0) \tag{5.6}$$

$$\begin{aligned}(\neg Holds(Carregada, s) \wedge Holds(Carregada, result(\alpha, s)) \rightarrow \\ \alpha = carregar\end{aligned} \tag{5.7}$$

$$\begin{aligned}(Holds(Vivo, s) \wedge \neg Holds(Vivo, result(\alpha, s)) \rightarrow \\ \alpha = atirar \wedge Holds(Carregada, s)\end{aligned} \tag{5.8}$$

$$Holds(Carregada, s) \rightarrow Holds(Carregada, result(\alpha, s)) \tag{5.9}$$

$$\neg Holds(Vivo, s) \rightarrow \neg Holds(Vivo, result(\alpha, s)) \tag{5.10}$$

As fórmulas (5.3) e (5.4) representam leis de efeito das ações *carregar* e *atirar*, respectivamente. As fórmulas (5.5) e (5.6) representam as observações empreendidas na situação inicial s_0. (5.7) e (5.8) constituem os axiomas de fecho de explicação para esse domínio. As fórmulas (5.9) e (5.10) são chamadas de *leis de imutabilidade*[7] e formalizam a convenção de que a arma não se descarrega e de que uma vítima morta não ressuscita.

Considerando-se a descrição do domínio acima, a partir de (5.3)–(5.10), pode-se inferir:

$$\neg Holds(Vivo, result(atirar, result(esperar, result(carregar, s_0)))). \tag{5.11}$$

■

Esse exemplo ilustra, de uma maneira bem simples, como a abordagem baseada em fecho de explicação pode ser empregada para solucionar o problema da persistência.

[7]No entanto, em [63, cap. 8], as fórmulas (5.9) e (5.10) são consideradas como axiomas de fecho de explicação. Preferimos a classificação acima por estar mais de acordo com a proposta original em [61].

O próximo exemplo ilustra a aplicação de axiomas de fecho de explicação a cenários envolvendo ramificações.

Exemplo 5.4.2 (Seqüência do exemplo 4.4.1) A formalização do cenário da caça ao peru no cálculo de situações com fecho de explicação é ilustrada abaixo:

$$Holds(Caminhando, s) \rightarrow Holds(Vivo, s) \tag{5.12}$$

$$Holds(Carregada, result(carregar, s)) \tag{5.13}$$

$$Holds(Carregada, s) \rightarrow \neg Holds(Vivo, result(atirar, s)) \tag{5.14}$$

$$Holds(Caminhando, s_0) \tag{5.15}$$

$$\neg Holds(Carregada, s_0) \tag{5.16}$$

$$\begin{aligned} &Holds(Caminhando, s) \wedge \neg Holds(Caminhando, result(\alpha, s)) \rightarrow \\ &\qquad (\alpha = atirar \wedge Holds(Carregada, s)) \end{aligned} \tag{5.17}$$

$$Holds(Carregada, s) \rightarrow Holds(Carregada, result(\alpha, s)) \tag{5.18}$$

$$\neg Holds(Vivo, s) \rightarrow \neg Holds(Vivo, result(\alpha, s)) \tag{5.19}$$

A fórmula (5.12) constitui a restrição de domínio relacionando *Caminhando* e *Vivo*. (5.13) e (5.14) formalizam as leis de efeito das ações *carregar* e *atirar*, respectivamente. As fórmulas (5.15) e (5.16) denotam as observações feitas na situação inicial s_0. (5.17) é o axioma de fecho de explicação para este cenário e (5.18) e (5.19) são como no exemplo anterior. A partir de (5.12)–(5.19), pode-se inferir a ramificação

$$\neg Holds(Caminhando, result(atirar, result(esperar, result(carregar, s_0))))$$

Por outro lado, observe que para essa descrição de domínio, não é possível a determinação da qualificação implícita da ação *atrair*, tal como nos exemplos 3.4.1 e 4.4.1. Isso só pode ser obtido mediante o acréscimo do seguinte axioma de fecho de explicação:

$$\begin{aligned} &(\neg Holds(Caminhando, s) \wedge Holds(Caminhando, result(\alpha, s)) \rightarrow \\ &\qquad \alpha = atrair \wedge Holds(Vivo, s) \end{aligned} \tag{5.20}$$

■

Na próxima seção, teremos a oportunidade de discorrer acerca de alguns pontos positivos da abordagem aqui apresentada, sem deixar de comentar, porém, sobre os principais problemas que apresenta.

5.5 Discussão crítica

O método apresentado neste capítulo, por ser baseado numa sintaxe enxuta, numa semântica não-inercial (contrariando a linha-guia das pesquisas em raciocínio sobre ações) e por ser fundamentado numa implementação eficiente e largamente utilizada, apresenta-se como uma alternativa realmente viável aos métodos não-monotônicos.

Com relação à complexidade do formalismo, temos que o aumento de complexidade resultante da axiomatização da persistência em axiomas de fecho de explicação é no máximo linear no número de axiomas de efeito [61]. Conforme apontado no referido trabalho, grosso modo, a adição de axiomas de fecho de explicação consiste em substituir os "se então" por "se e somente se". Nesse caso, a adição de axiomas representando cada "somente se" aumenta a complexidade apenas por um fator constante.

Entretanto, essa é apenas uma visão simplificada do processo, pois a abordagem de fecho de explicação não se baseia em simples bicondicionalizações dos axiomas de efeito. Na verdade, tal simplificação é colocada apenas para ilustrar o limite superior do aumento de complexidade produzido pelos axiomas de fecho de explicação.

A idéia geral dos axiomas de fecho de explicação foi posteriormente utilizada por Reiter [56]. No entanto, este considera a *hipótese generalizada de completude*[8] e codifica os axiomas de fecho de explicação na forma de *bicondicionais*, ao contrário da idéia original, segundo a qual os mesmos são codificados na forma de *condições necessárias*.

No trabalho de Reiter, os axiomas de fecho de explicação são derivados automaticamente a partir das leis de efeito das ações e combinados com estas em bicondicionais, com o objetivo de reduzir a si mesmos a explicação de *todas* as possibilidades com que as mudanças podem ocorrer. Como exemplo, em tal abordagem, o fato $Próximo(Robô, x)$ torna-se verdadeiro *se e somente se* o robô em questão executar a ação $irpara(x)$.

Entretanto, o método proposto em [56] produz condições extremamente fortes que, em determinadas circunstâncias, podem não condizer com a realidade. Como exemplo, temos a situação em que o fato de o robô ficar próximo a x é um efeito indireto de ele ter executado a ação $irpara(y)$, estando x próximo a y [62].

Para corrigir esses inconvenientes, o mesmo cenário deve ser representado de modo que $Próximo(Robô, x)$ seja verdadeiro *somente se* o robô executar $irpara(y)$, para algum y, tal que $x = y$ ou $Próximo(x, y)$ [62]. Note, porém, que tal representação envolve ainda a utilização de alguma forma de se expressar o fecho transitivo do predicado *Próximo*.

Uma crítica que poderia ser posta à abordagem aqui apresentada é o fato de a contrapositiva de cada axioma de fecho de explicação se parecer com um axioma de persistência.

[8] No original em inglês, *generalized completeness assumption* — *GCA*.

Entretanto, conforme apontado em [61], basta observar que os axiomas de fecho de explicação expressam não-mudança em termos de *não-ocorrência* de certas ações, e não em termos de *ocorrência*, como fazem os axiomas de persistência.

Em outros termos, enquanto um axioma de persistência enuncia que uma dada propriedade não se altera após a *execução* de uma dada ação, a contrapositiva de um axioma de fecho de explicação tem por objetivo permitir a dedução da persistência de tal propriedade devido à *não execução* de tal ação. A diferença, apesar de sutil, é muito importante para se poder enxergar os axiomas de fecho de explicação como solução efetiva ao problema da persistência.

Conforme destacado em [62], a utilização de axiomas de fecho de explicação, por ser bastante sucinta, oferece uma alternativa viável aos métodos de circunscrição e às abordagens não-monotônicas em geral. Entretanto, em [13] é proposta uma versão circunscritiva de fecho de explicação, baseada numa modificação de PMON[9] [59], que, além de apresentar um método de prova eficiente baseado em circunscrição para certas classes restritas de problemas, permite ainda a geração automática dos axiomas de fecho de explicação.

Nesse trabalho, os autores afirmam que os argumentos contra métodos não-monotônicos não são fundamentados, uma vez que, segundo eles, ainda não estão bem claras as fronteiras reais entre abordagens monotônicas e não-monotônicas. No entanto, o método circunscritivo proposto ainda cria problemas sutis durante a fase de representação do domínio e a determinação de sua política de circunscrição não é trivial.

Um aspecto negativo do método aqui apresentado diz respeito à completude, pois sua implementação baseada em STRIPS usando a estratégia do cão dorminhoco não é completa para todas as classes de teorias de ações [61].

O problema maior decorrente disso é o fato de que qualquer tentativa de aumentar o conjunto de teorias às quais essa abordagem possa ser aplicada diminuiria consideravelmente a eficiência de seu método de prova para o caso geral.

Com relação à questão da hipótese de fecho de ações, em [62, seção 2], o autor discorre sobre as diferenças fundamentais, embora sutis, de sua proposta semântica para o uso de axiomas de fecho de ações em relação às comumente encontradas na literatura.

Segundo ele, há uma grande diferença entre se conhecer todas as ações possíveis de um dado domínio em representação (onisciência divina) e se conhecer todas as ações realmente relevantes e passíveis de serem executadas em um dado contexto[10]. Com isso,

---

[9] *Pointwise minimization of occluded with nonchange premisses.*

[10] Um exemplo ilustrativo disso é uma situação na qual um pintor, ao entrar numa sala, precisa apenas saber que as ações que podem ser realizadas são entrar na sala, pintar a mesma, e sair dela. Não interessa para o pintor, neste contexto restrito, saber que podem ocorrer as ações de jogar um avião nas torres do World Trade Center ou de implodi-las (a menos, é claro, que ele esteja numa de suas salas!).

a abordagem em [62] utiliza uma versão um pouco mais relaxada da hipótese de fecho de ações do que a implicitamente definida em [61] e em outros formalismos.

Cabe, agora, uma pequena discussão filosófica a respeito do método analisado neste capítulo. Olhando bem para a estrutura de um axioma de fecho de explicação, fica claro que o raciocínio subjacente é muito próximo de uma inferência abdutiva. Como exemplo, a partir do axioma de fecho de explicação

$$(\neg Holds(Carregada, s) \wedge Holds(Carregada, result(\alpha, s)) \rightarrow \alpha = carregar \qquad (5.21)$$

e, observando-se que na situação s é executada a ação *carregar*, conclui-se

$$\neg Holds(Carregada, s) \wedge Holds(Carregada, result(\alpha, s))$$

Note a semelhança (acidental?) entre o axioma de fecho de explicação (5.21) e a regra (não-válida) de abdução:

$$\frac{A \rightarrow B, B}{A}$$

Resta, assim, uma pergunta: até que ponto tal forma de raciocínio é realmente válida? A resposta, acreditamos, está na hipótese de fecho de ações, que garante que se sabe tudo sobre as mudanças que podem ocorrer em um dado contexto.

Mas, mesmo assim, até que ponto tal hipótese resiste à existência de outras possibilidades de mudanças no domínio, uma vez que o próprio Schubert considera as abordagens não-monotônicas muito fortes e clama por uma maior flexibilidade de representação? Até que ponto isso pode influenciar em circunstâncias nas quais o aprendizado de novas ações pode ser interessante? Ao que tudo indica, tais questões continuam ainda em aberto.

Outra questão fundamental seria: até que ponto a abordagem de fecho de explicação pode realmente ser considerada como monotônica? Nada impede que os axiomas de fecho de explicação sejam interpretados como uma espécie de negação por falha [11]. Nesse caso, estando essa idéia embutida na própria lógica de base, propagando-se indiretamente pelo método de prova, conclui-se que o formalismo aqui apresentado não é de todo monotônico, ao menos não na acepção do termo.

Seja como for, entretanto, Schubert procura deixar claro que não pressupõe que os métodos monotônicos são tudo de que se precisa para resolver todos os problemas em raciocínio sobre ações. Tanto, que, como caminho a ser trilhado para a obtenção de uma solução ao problema da qualificação, sugere o uso de uma abordagem probabilística [62].

Na verdade, conforme colocado no início dessa seção, o autor do fecho de explicação procura apenas mostrar que os principais problemas para os quais os métodos não-monotônicos apresentam dificuldades podem ser resolvidos monotonicamente.

Uma crítica bastante importante comumente feita ao uso de fecho de explicação diz respeito à necessidade de se enunciar "na mão" todos os axiomas de fecho de explicação do domínio. Sendo assim, algumas propostas procuram automatizar essa tarefa [56, 13].

Entretanto, qualquer tentativa de gerar axiomas de fecho de explicação de uma maneira completamente automática pode acabar levando a resultados não-intuitivos. Isso se deve, em parte, ao fato de a determinação dos axiomas de fecho de explicação ser bastante dependente da semântica de base do domínio em consideração, o que faz com que sua codificação tenha de ser supervisionada. Retornaremos à questão da geração de descrições do domínio no decorrer deste trabalho.

No próximo capítulo, analisaremos outra abordagem monotônica para raciocinar sobre ações cuja base semântica lembra em muito os axiomas de fecho de explicação, apresentando, porém, uma noção causal baseada em termos de dependência.

Capítulo 6

Relações de dependência

O curso das nuvens não depende do latido de um cão.

— Provérbio árabe

Neste capítulo, analisaremos a abordagem apresentada por Castilho e colegas [7], a qual também consiste em um formalismo monotônico para se raciocinar sobre ações.

Tal proposta ilustra muito bem a aplicação de lógicas não-clássicas em raciocínio sobre ações e se caracteriza pela definição de uma noção causal explícita representada por meio de uma relação de dependência entre ações e literais.

Utilizando relações de dependência, pode-se resolver certas classes de problemas que não são corretamente tratadas por outras abordagens presentes na literatura. Além disso, o formalismo aqui apresentado permite ainda a definição de um método de prova de implementação viável.

6.1 Fundamentação

A idéia essencial desta abordagem é definir uma lógica de base fundamentada em lógica modal, acrescentando a esta uma noção de causalidade explícita e lhe adaptando um método de prova adequado e completo. No restante deste capítulo, suporemos que o leitor conhece os rudimentos da lógica modal [32, 10, 33] e dos sistemas de tableau [18, 25].

O formalismo de base utilizado pela abordagem baseada em relações de dependência é a *lógica de ações e planos* $\mathcal{LAP}$, que basicamente consiste de uma lógica multimodal proposicional que demonstra possuir um poder de representação bastante expressivo [4].

Apesar de ser perfeitamente possível a utilização do cálculo de situações como substrato lógico para esta abordagem, preferimos seguir a formulação original do referido trabalho. As vantagens de se preferir lógica modal ao cálculo de situações ficarão mais evidentes ao longo deste e dos próximos capítulos.

A noção de causalidade utilizada neste formalismo é representada através de uma *relação de dependência* envolvendo ações e literais, a qual especifica quais ações podem afetar determinados literais. Sendo assim, esta abordagem causal também se caracteriza por *permitir* mudança, e não por *forçá-la*. Com isso, a relação de dependência assim definida constitui uma *conexão causal fraca* entre ações e literais.

Dizer que um certo literal L *depende* de uma dada ação α significa que α *pode causar* L, ou seja, após a execução de α, L *pode* se tornar verdadeiro. Por outro lado, dizer que L é *independente* de α significa que, executando-se α, seu valor *permanece* o mesmo. Se não estiver explícito que L depende de α, então considera-se que L é independente de α.

É exatamente dessa maneira que pode-se obter uma solução para o problema da persistência. Como, pela hipótese da inércia, ao se executar uma determinada ação, a maioria dos literais permanece inalterada, conclui-se que a grande maioria dos literais é independente da ação em questão. Isso significa que, ao invés de se escreverem todos os axiomas de persistência necessários envolvendo tais literais, basta enunciar que os mesmos são independentes, o que equivale a dizer que os literais restantes são dependentes.

Utilizando o raciocínio acima esboçado, os modelos possíveis da lógica de base $\mathcal{LAP}$ são restritos por meio de uma condição, baseada na dependência, que estabelece que toda vez que um dado literal L for independente de uma ação α, o seu valor de verdade permanece inalterado após a execução de α, o que faz com que sua não-mudança seja derivada diretamente, sem a necessidade de axiomas de persistência. Maiores informações acerca dos detalhes envolvidos nesse procedimento serão apresentados na seqüência.

Na próxima seção, apresentaremos a definição da lógica de ações e planos $\mathcal{LAP}$, que constitui a lógica de base sobre a qual se fundamenta o método aqui analisado e que será usada no restante deste trabalho.

6.2 Lógica de base

Apresentamos nesta seção a formalização da *lógica de ações e planos* $\mathcal{LAP}$ [4], a qual consiste basicamente numa simplificação da lógica dinâmica proposicional PDL[1] [28]. Em linhas gerais, $\mathcal{LAP}$ é uma lógica multimodal proposicional, contendo um operador modal $[\alpha]$ para cada ação atômica α, e um operador modal suplementar $\Box$ para representar leis do domínio, ou seja, fórmulas que são sempre verdadeiras, independente da situação.

A lógica de cada operador $[\alpha]$ é dada pela do sistema de lógica modal K, enquanto que a do operador $\Box$ é a do sistema $S4$. Para mais detalhes acerca dos sistemas de lógica modal, recomenda-se [10].

[1]No original em inglês, *Propositional Dynamic Logic*.

Define-se $A\tilde{C}\tilde{O}ES = \{\alpha, \beta, \ldots\}$ como o conjunto formado por *ações atômicas*, e.g. *atirar* e *carregar*. Define-se também $ATM = \{P, Q, \ldots\}$ como o conjunto das *fórmulas atômicas*, ou simplesmente *átomos*, como, por exemplo, *Carregada* e *Vivo*.

Dados os dois conjuntos acima, define-se ainda $LIT = ATM \cup \{\neg P : P \in ATM\}$ como o conjunto dos *literais*, ou seja, todos os átomos e suas respectivas negações. Define-se também o conjunto de todas as *fórmulas* FOR como sendo o menor conjunto tal que:

- $ATM \subseteq FOR$;
- $\top, \bot \in FOR$, onde $\top$ representa a *tautologia* e $\bot$ a *contradição*;
- Se $A, B \in FOR$ e $\alpha \in A\c{C}\tilde{O}ES$, então $\neg A, A \wedge B, \Box A, [\alpha]A \in FOR$;

As expressões $A \vee B$, $A \rightarrow B$ e $A \leftrightarrow B$ constituem abreviaturas tal qual usualmente definidas. As fórmulas $\langle\alpha\rangle A$ e $\Diamond A$ são abreviaturas para $\neg[\alpha]\neg A$ e $\neg\Box\neg A$, respectivamente, e serão utilizadas com esse fim.

Uma fórmula do tipo $[\alpha]A$ é lida como "A é verdadeira após a execução da ação α", e uma do tipo $\Box A$ como "A é sempre verdadeira". Em particular, $[\alpha]\bot$ significa "α não é executável". Lê-se $\langle\alpha\rangle\top$ como "α é executável" e $\langle\alpha\rangle A$ como "existe uma execução possível de α que produz A". Já $\Diamond A$ será utilizada para tarefas de planificação e sua leitura será "existe uma seqüência de ações que torna A verdadeira". Se uma fórmula não possui operador modal, então ela é dita como sendo uma fórmula *clássica*.

Define-se as *leis de domínio* como um conjunto finito formado por fórmulas da forma $\Box A$, as quais são lidas como "A é sempre verdadeira". De acordo com a forma da subfórmula A, divide-se tal conjunto em duas categorias: *leis estáticas* e *leis dinâmicas*.

Se uma fórmula $\Box A$ é tal que sua subfórmula A é uma fórmula clássica, ou seja, não contém operador de ação $[\alpha]$, para algum $\alpha \in A\c{C}\tilde{O}ES$, então $\Box A$ é considerada uma *lei estática*, ou *restrição de domínio*. Um exemplo de lei estática é

$$\Box(Caminhando \rightarrow Vivo).$$

Leis estáticas servem para dizer que determinadas fórmulas são sempre verdadeiras, independente de quaisquer ações. O conjunto de todas as leis estáticas presentes em uma determinada descrição de um mundo será representado por $ESTAT$.

Se, por outro lado, uma fórmula $\Box A$ apresentar ao menos um operador modal $[\alpha]$, com $\alpha \in A\c{C}\tilde{O}ES$, então tal fórmula é uma *lei dinâmica*. Leis dinâmicas consistem de fórmulas que representam ações e seus efeitos. Exemplos de leis dinâmicas são

$$\Box[carregar]Carregada \tag{6.1}$$

$$\Box(\neg Vivo \rightarrow [atirar]\neg Vivo) \tag{6.2}$$

A lei dinâmica (6.1) estabelece que "após se executar a ação de carregar uma arma, a mesma estará carregada". A lei (6.2), por sua vez, traduz o fato de que "se a vítima está morta, então após a ação de atirar ela continuará morta".

Tais fórmulas podem ser formadas por uma ou mais ações e são ainda classificadas em duas categorias, as *leis de executabilidade das ações* e as *leis dos efeitos das ações*.

As leis de executabilidade das ações consistem de fórmulas da forma $\Box(B \rightarrow \langle\alpha\rangle\top)$. Tais fórmulas servem para estabelecer as condições sob as quais as ações podem ser executadas. Como exemplo, a lei

$$\Box(\mathit{PossuiArma} \rightarrow \langle \mathit{atirar} \rangle \top)$$

significa que, uma vez de posse de uma arma, pode-se atirar. Um caso particular deste tipo de lei é quando a condição B é $\top$. Nesse caso, tem-se que a ação é sempre executável. O conjunto de todas as leis de executabilidade de um domínio será denotado por EXE.

As leis dos efeitos das ações são fórmulas da forma $\Box(B \rightarrow [\alpha]C)$, onde B e C são fórmulas clássicas. Fórmulas desse tipo são lidas como "é sempre verdade que, se B for verdadeira, então após a execução de α, C é verdadeira". B, neste caso, é a *pré-condição* da ação α, e C é a *pós-condição*. Como exemplo de uma lei de efeitos de ações, tem-se

$$\Box(\mathit{Carregada} \rightarrow [\mathit{atirar}]\neg \mathit{Vivo}).$$

Nesse caso, tal fórmula representa que toda vez que se executar a ação *atirar* com uma arma que esteja previamente carregada, a vítima morrerá. O conjunto de todas as leis dos efeitos das ações como a desse exemplo será denotado por LEA.

Algumas leis de efeito, dada a forma que apresentam, podem receber nomes especiais:

- *Axiomas de persistência condicionais*, que são fórmulas da forma $\Box((C \wedge B) \rightarrow [\alpha]B)$. Representam o fato de que, dada uma certa condição C, a execução da ação α mantém inalterado o valor de verdade de B. Um exemplo bastante típico é

  $$\Box((\neg \mathit{Carregada} \wedge \mathit{Vivo}) \rightarrow [\mathit{atirar}]\mathit{Vivo}),$$

  que denota que, tendo em mãos uma arma descarregada, é impossível matar alguém.

  Um caso particular desta categoria é quando a condição C é $\top$, resultando nos chamados axiomas de persistência. A este tipo de lei dinâmica está associado o problema da persistência, conforme discutido na seção 2.2.

- *Leis de inexecutabilidade*, que são fórmulas da forma $\Box(B \rightarrow [\alpha]\bot)$. De acordo com o próprio nome, este tipo de lei dinâmica representa o fato de que, em função da condição B, a ação α não pode ser executada. Como exemplo, temos

$$\Box(\neg PossuiArma \rightarrow [atirar]\bot),$$

que significa que não estando de posse de uma arma, não se pode atirar.

Define-se uma *teoria de ações*, denotada por *LEIS*, como o conjunto formado por todas as leis estáticas, leis de efeitos das ações e leis de executabilidade das ações:

$$LEIS = ESTAT \cup LEA \cup EXE.$$

Uma *base de conhecimento BC* é definida como o conjunto de observações sobre um dado mundo em um dado instante. Em essência, BC consiste de um conjunto finito de fórmulas sem o operador modal $\Box$. BC pode conter ainda fórmulas que expressam informações sobre o estado do mundo após a execução de uma determinada seqüência de ações. Um exemplo de base de conhecimento seria

$$BC = \{Caminhando, Vivo, [atirar]\,Vivo\},$$

que representa um mundo no qual foi observado que um peru estava vivo, caminhando e que após alguém ter-lhe atirado ele não morreu.

Para a lógica de ações e planos $\mathcal{LAP}$, é definida uma semântica baseada na *semântica dos mundos possíveis* de Kripke [34]:

Definição 6.2.1 Define-se um modelo para $\mathcal{LAP}$ ($\mathcal{LAP}$-modelo) como uma quádrupla $\mu = \langle W, \{R_\alpha : \alpha \in A\c{C}\tilde{O}ES\}, R_\Box, \tau\rangle$, na qual W é um conjunto de mundos possíveis, $R_\Box$ e cada R_α são relações binárias sobre W (chamadas de relações de acessibilidade) e $\tau : ATM \longrightarrow 2^W$ é uma interpretação para os átomos (ou valoração). $R_\Box$ deve ser *reflexiva* e *transitiva*, e $R_\alpha \subseteq R_\Box$, para toda ação $\alpha \in A\c{C}\tilde{O}ES$. ■

As condições de verdade para as fórmulas clássicas de $\mathcal{LAP}$ são definidas da maneira usual. Para as fórmulas com operadores modais, temos as seguintes condições de verdade:

- $\models^\mu_w \Box A$ se e somente se para todo $w' \in W, wR_\Box w'$ implica $\models^\mu_{w'} A$;
- $\models^\mu_w [\alpha]A$ se e somente se para todo $w' \in W, wR_\alpha w'$ implica $\models^\mu_{w'} A$;
- $\models^\mu_w \Diamond A$ se e somente se existe $w' \in W$, tal que $wR_\Box w'$ e $\models^\mu_{w'} A$;
- $\models^\mu_w \langle\alpha\rangle A$ se e somente se existe $w' \in W$, tal que $wR_\alpha w'$ e $\models^\mu_{w'} A$.

Uma fórmula A é *verdadeira* em um $\mathcal{LAP}$-modelo $\mu = \langle W, \{R_\alpha : \alpha \in A\c{C}\tilde{O}ES\}, R_\Box, \tau\rangle$ se $\models^\mu_w A$ para todo mundo possível $w \in W$. A é $\mathcal{LAP}$-válida, denotado $\models_{\mathcal{LAP}} A$, se A

é verdadeira em todo $\mathcal{LAP}$-modelo. A é $\mathcal{LAP}$-satisfatível se $\models_w^\mu A$ para algum mundo possível $w \in W$ de algum modelo $\mu = \langle W, \{R_\alpha : \alpha \in A\c{C}\tilde{O}ES\}, R_\Box, \tau\rangle$.

Sejam BC_1 e BC_2 duas bases de conhecimento contendo observações a respeito de duas circunstâncias em particular, e seja $\alpha = \alpha_1; \ldots; \alpha_n$ uma seqüência de ações. Em $\mathcal{LAP}$, pode-se realizar as tarefas de predição, explicação e planificação, definidas na seção 1.2, provando-se, respectivamente, as seguintes fórmulas:

$$(LEIS \wedge BC_1) \rightarrow \langle\alpha\rangle BC_2 \tag{6.3}$$

$$(LEIS \wedge \langle\alpha\rangle BC_2) \rightarrow BC_1 \tag{6.4}$$

$$(LEIS \wedge BC_1) \rightarrow \Diamond BC_2 \tag{6.5}$$

A fórmula (6.3) representa que, dados um conjunto de leis e um estado do mundo BC_1, após a execução de α teremos um estado do mundo BC_2. A fórmula (6.4) expressa que dados um conjunto de leis e o fato de que após executar-se α obteve-se um estado do mundo BC_2, então antes da execução de α o estado do mundo era dado por BC_1. A fórmula (6.5), por sua vez, representa que, dados um conjunto de leis e um estado do mundo BC_1, é possível obter-se um estado do mundo correspondente a BC_2.

Importante ressaltar, entretanto, que em $\mathcal{LAP}$ o problema da persistência não está resolvido, de maneira que ainda se faz necessário o enunciado de axiomas de persistência para a representação correta de um dado domínio. Na próxima seção, descrevemos o método proposto para se evitar de escrever axiomas de persistência no conjunto de leis e ainda poder raciocinar sobre ações em domínios mais elaborados.

6.3 Método empregado

Dada uma descrição de domínio contendo todas as leis de efeito e todos os axiomas de persistência, para cada ação α e literal L, a presença de um axioma de persistência da forma $\Box(L \rightarrow [\alpha]L)$ indica que a ação α *não pode* alterar o valor de verdade de L de verdadeiro para falso, qualquer que seja a situação considerada. Isso equivale a dizer, para este caso, que α *não pode causar* $\neg L$, ou, simplesmente, $\neg L$ *é independente de* α.

Usando essa observação como base, é definida uma relação de independência entre ações e literais, contendo pares da forma $(\alpha, \neg L)$, denotando que a execução da ação α não altera o valor de verdade de L. Como cada axioma de persistência $\Box(L \rightarrow [\alpha]L)$ determina um e somente um par $(\alpha, \neg L)$ na independência e vice versa, é lícito afirmar que se pode substituir o conjunto de todos os axiomas de persistência pela relação de independência, obtendo-se, assim, a mesma informação, apenas expressa de maneira diferente.

No entanto, observando mais atentamente, percebe-se que, no final das contas, acaba-se trocando um potencialmente grande conjunto de axiomas de persistência por uma igualmente grande relação de independência. Entretanto, como a hipótese da inércia estabelece que aquilo que muda em função da execução de uma ação α é muito pouco, tudo leva a crer que a grande maioria dos literais deve ser independente de α. Assim, é razoável esperar que o complementar da relação de independência seja muito menor do que esta. Com isso, a idéia consiste em se representar a relação de independência por meio de seu complementar, a *relação de dependência*, cuja definição é dada a seguir:

Definição 6.3.1 Uma *relação de dependência* é uma relação binária $\leadsto \subseteq AÇ\tilde{O}ES \times LIT$. O complementar de $\leadsto$ é denotado $\not\leadsto$ e é chamado de *relação de independência*. ■

A expressão $\alpha \leadsto L$ significa que "α *pode causar* L", representando o fato de que a execução da ação α *pode* alterar o valor de verdade de L de falso para verdadeiro. Como exemplo, seja a dependência dada por $atirar \leadsto \neg Vivo$, que denota que a execução de *atirar* pode fazer com que o valor de verdade de $\neg Vivo$ mude de falso para verdadeiro. Assim, a noção causal subjacente à relação de dependência não necessariamente *causa* mudança, mas, pelo contrário, apenas a *permite*, o que possibilita um formalismo mais flexível.

Outro ponto importante é o fato de que, ao contrário da minimização da causalidade, analisada no capítulo 3, e do fecho de explicação, sobre o qual discorremos no capítulo 5, a noção causal representada por $\leadsto$ não está embutida na lógica de base, mas sim no nível de uma metalinguagem, de forma semelhante à proposta de pós-derivação de efeitos, que tivemos a oportunidade de analisar no capítulo 4.

Dada uma relação de dependência $\leadsto$, integrando-a à lógica definida na seção anterior, obtém-se $\mathcal{LAP}_{\leadsto}$, uma nova lógica baseada em dependência [4].

Os modelos de $\mathcal{LAP}_{\leadsto}$ são exatamente os modelos de $\mathcal{LAP}$ que satisfazem a condição de dependência estabelecida. Tal condição enuncia que se a execução de uma dada ação α em um mundo possível w leva ao mundo w', então para cada literal L tal que $\alpha \not\leadsto \neg L$, se L é verdadeiro em w, então L também será verdadeiro em w'.

Definição 6.3.2 Seja $\leadsto$ uma relação de dependência entre ações e literais. Um modelo de $\mathcal{LAP}_{\leadsto}$ é um $\mathcal{LAP}$-modelo $\mu = \langle W, \{R_\alpha : \alpha \in AÇ\tilde{O}ES\}, R_\Box, \tau \rangle$, tal que, sempre que $wR_\alpha w'$, teremos que para todo $\alpha \in AÇ\tilde{O}ES$:

(a) $\alpha \not\leadsto L$, $w \notin \tau(L)$ implica $w' \notin \tau(L)$;

(b) $\alpha \not\leadsto \neg L$, $w \in \tau(L)$ implica $w' \in \tau(L)$. ■

Em outros termos, um $\mathcal{LAP}_{\leadsto}$-modelo é um $\mathcal{LAP}$-modelo com uma condição suplementar para tratar dependências entre ações e literais: se α não pode causar L (resp. $\neg L$), então a execução de α deixará inalterado o valor de verdade de $\neg L$ (resp. L).

A figura 6.1 a seguir ilustra as condições (a) e (b) da definição 6.3.2.

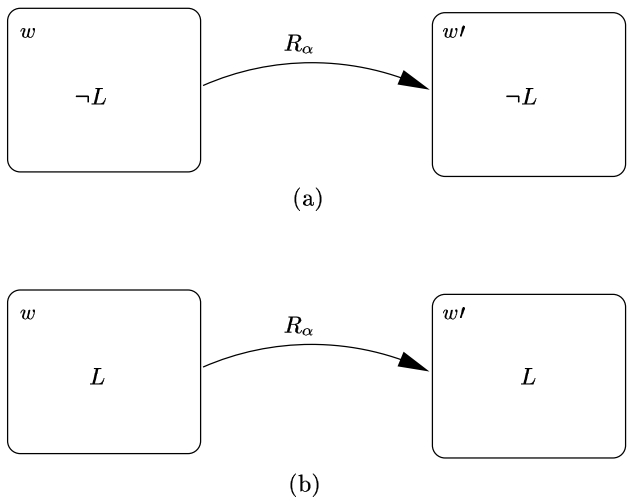

Figura 6.1: Ilustração de transição entre mundos, considerando as condições baseadas em dependência (a) e (b) dadas na definição 6.3.2.

Dada uma relação de dependência $\leadsto$, diz-se que uma fórmula A é $\mathcal{LAP}_{\leadsto}$-válida (denotado por $\models_{\mathcal{LAP}_{\leadsto}} A$) se A for verdadeira em todos os $\mathcal{LAP}_{\leadsto}$-modelos.

Em adição à axiomática de $\mathcal{LAP}$ [4], deve-se enunciar ainda para $\mathcal{LAP}_{\leadsto}$ o seguinte esquema de axioma, que estabelece a restrição determinada pela relação de dependência:

$$Persistence([\alpha]) : L \rightarrow [\alpha]L,\ \text{se}\ \alpha \not\leadsto \neg L$$

O fato de $\mathcal{LAP}$ ser uma lógica compacta e completa [7] lhe possibilita a adaptação de um método de prova baseado nos sistemas de tableau semânticos nos moldes de [5].

A grande vantagem dos sistemas de tableau reside no fato de constituírem métodos de prova adequados e completos, propriedade esta que nenhuma outra abordagem satisfatória para os problemas em raciocínio sobre ações apresentou até o momento.

Omitiremos aqui a definição do método de tableau para $\mathcal{LAP}_{\leadsto}$, justamente por tratar-se de um caso particular do método que será apresentado no capítulo 8.

Na próxima seção, utilizaremos $\mathcal{LAP}_{\leadsto}$ na representação de domínios, ilustrando a maneira como esse formalismo trata os problemas da persistência e da ramificação.

6.4 Cenários-exemplos

Apresentamos nesta seção dois exemplos de aplicação de $\mathcal{LAP}_{\leadsto}$ como formalismo de base para a realização de tarefas em raciocínio sobre ações. O primeiro consiste do clássico cenário do tiro em Yale, que bem ilustra a maneira como a abordagem analisada neste capítulo resolve o problema da persistência:

Exemplo 6.4.1 (Seqüência do exemplo 5.4.1)

$$LEIS = \left\{ \begin{array}{l} \Box\langle esperar\rangle\top, \\ \Box\langle carregar\rangle\top, \\ \Box\langle atirar\rangle\top, \\ \Box[carregar]\,Carregada, \\ \Box[atirar]\neg Carregada, \\ \Box(Carregada \rightarrow [atirar]\neg Vivo), \\ \Box((\neg Carregada \wedge Vivo) \rightarrow [atirar]\,Vivo) \end{array} \right\}$$

$$\leadsto\; = \left\{ \begin{array}{l} atirar \leadsto \neg Carregada, \\ atirar \leadsto \neg Vivo, \\ carregar \leadsto Carregada \end{array} \right\}$$

$$BC = \{\neg Carregada, Vivo\}$$

A partir dessa modelagem, pode-se inferir:

$$\models_{\mathcal{LAP}_{\leadsto}} (BC \wedge LEIS) \rightarrow [carregar][esperar][atirar](\neg Carregada \wedge \neg Vivo).$$

A independência $esperar \not\leadsto \neg Carregada$ garante que a arma continua carregada após a execução de *esperar*.

Observe, porém, a presença do axioma de persistência condicional

$$\Box((\neg Carregada \wedge Vivo) \rightarrow [atirar]\,Vivo).$$

Retornaremos a esse ponto na discussão crítica da próxima seção. ■

O próximo exemplo é uma instância de problemas envolvendo tanto efeitos indiretos como qualificações implícitas de ações:

Exemplo 6.4.2 (O cenário da porta bloqueada [7]) Para este cenário, consideram-se duas ações possíveis: *abrir*, que tem o efeito direto de abrir uma porta, caso esta não esteja bloqueada, e *arrombar*, mais forte do que a primeira e cujo efeito direto é abrir a porta, independente das circunstâncias. A ação *abrir* possui a qualificação implícita de que a porta precisa estar desbloqueada para que ela possa ser executada, enquanto que *arrombar* produz o efeito indireto de desbloquear a porta. A representação deste cenário em $\mathcal{LAP}_{\leadsto}$ é dada a seguir:

$$LEIS = \left\{ \begin{array}{l} \Box(Bloqueada \rightarrow Fechada), \\ \Box[abrir]\neg Fechada, \\ \Box[arrombar]\neg Fechada \end{array} \right\}$$

$$\leadsto\; = \left\{ \begin{array}{l} abrir \leadsto \neg Fechada, \\ arrombar \leadsto \neg Fechada, \\ arrombar \leadsto \neg Bloqueada \end{array} \right\}$$

$$BC = \{Fechada, Bloqueada\}$$

A partir dessa representação, pode-se inferir

$$\models_{\mathcal{LAP}_{\leadsto}} (BC \wedge LEIS) \rightarrow [arrombar](\neg Fechada \wedge \neg Bloqueada),$$

sendo possível, portanto, derivar o efeito indireto da ação *arrombar*.

Da mesma forma, é possível mostrar que

$$\models_{\mathcal{LAP}_{\leadsto}} (BC \wedge LEIS) \rightarrow [abrir]\bot.$$

Isso significa que com $\mathcal{LAP}_{\leadsto}$ é possível derivar também a qualificação implícita para a ação *abrir*: ela não pode ser executada se a porta estiver bloqueada. Note, porém, que neste exemplo não foram enunciadas as leis de executabilidade das ações, uma vez que sua presença tornaria esse cenário inconsistente. ■

Na próxima seção, traçaremos alguns comentários acerca da abordagem analisada neste capítulo, destacando os ganhos que a comunidade teve com seu formalismo, bem como os pontos que ainda precisam ser melhorados.

6.5 Discussão crítica

Um consenso na comunidade de raciocínio sobre ações é a idéia de que todo estudo sistemático de ações deve ser baseado em um formalismo o mais simples possível. Isso tem sido apoiado pelos trabalhos realizados na última década, em que ficou evidente que o cálculo de situações, neste sentido, é rico demais para os propósitos desejados.

Seguindo essa linha, a abordagem baseada em relações de dependência apresenta um formalismo que além de simples é bastante elegante e flexível. Observando-se os resultados em [5, 4, 7], fica evidente que, a despeito de se utilizar uma lógica proposicional, em nenhum quesito esta fica devendo algo ao cálculo de situações em termos de expressividade.

Entretanto, caberia aqui uma dúvida: se o ponto chave da solução apresentada neste capítulo é justamente a relação de dependência, a qual é independente da sintaxe de base, então qual a vantagem em se usar uma lógica heterodoxa como $\mathcal{LAP}$? Por que não se utilizou o cálculo de situações (ou até mesmo outro formalismo clássico) como substrato lógico, visto que todos os operadores modais podem ser definidos como predicados?

De fato, existe a possibilidade de se utilizar outra lógica ao invés de $\mathcal{LAP}$ como base para essa abordagem. A resposta às perguntas acima consiste no fato de $\mathcal{LAP}$ ser decidível e possuir um método de prova adequado e completo. Além disso, tal método não somente é implementável, como já está implementado (veja, por exemplo, os trabalhos presentes em [67] e [14]). Sendo assim, a pergunta passa a ser: por que então utilizar uma lógica indecidível e para a qual sabe-se não haver métodos de prova satisfatórios e cuja implementação não seja uma realidade, se se pode conseguir tudo isso com $\mathcal{LAP}$?

Uma outra questão, entretanto, seria: ora, e por acaso não existem métodos de tableau para a lógica de predicados? Em resposta a isso, da mesma forma, apela-se para o critério da decidibilidade [3]. Além do mais, conforme ficará mais claro no capítulo 8, a introdução da condição de dependência em tableaux modais se faz mediante o acréscimo de regras de apenas um passo, enquanto que para lógicas predicativas não está claro como que tais modificações deveriam ser empreendidas[2].

No tocante à semântica existente por detrás da noção de dependência, a abordagem aqui apresentada é bastante próxima à idéia de fecho de explicação, discutida no capítulo 5. Basicamente, pode-se enxergar a dependência $atirar \leadsto \neg Vivo$ como explicitando que a única maneira pela qual $Vivo$ pode ter seu valor alterado de verdadeiro para falso é mediante a execução da ação $atirar$ [7].

Um ponto bastante delicado nesta abordagem, entretanto, é a necessidade de se enunciar axiomas de persistência condicionais. Observe que, no exemplo 6.4.1, foi necessária a inclusão do axioma de persistência condicional

$$\Box((\neg Carregada \land Vivo) \rightarrow [atirar]Vivo).$$

Sem a adição de axiomas dessa forma à representação do domínio, não seria possível obter as conclusões desejadas. É justamente a necessidade de inclusão de tais axiomas que constitui a maior crítica da comunidade à abordagem baseada em relações de dependência,

[2]A grande vantagem da lógica modal consiste no fato de seus métodos de tableau se adequarem perfeitamente à semântica dos mundos possíveis definida em [34]. Neste sentido, a aplicação das regras de tableau, conforme será ilustrado no capítulo 8, simula perfeitamente as transições entre os mundos possíveis (no nosso caso, estados possíveis do domínio representado).

uma vez que os demais formalismos presentes na literatura não se utilizam desse tipo de axioma para resolver o problema da persistência.

Entretanto, segundo apontado em [7], os axiomas de persistência condicionais consistem, na realidade, de uma conseqüência da utilização da hipótese de fecho de explicação, tal qual usada em [56]. A diferença fundamental da abordagem com $\mathcal{LAP}_{\rightsquigarrow}$ em relação a esta, no entanto, reside no fato de, tal como em [61], deixar a cargo do projetista a tarefa de adicionar tais axiomas à base, ao invés de gerá-los de maneira completamente automática. Com isso, tem-se a vantagem de manter a lógica de base inalterada.

Um exemplo bastante marcante das implicações de se utilizar uma ou outra de tais abordagens é dado pelo cenário a seguir:

Exemplo 6.5.1 (O cenário do tiro em Toulouse [20]) Seja a seguinte situação, semelhante à do cenário do tiro em Yale [27], na qual, porém, tem-se um revólver cujo cilindro contém no máximo seis balas. Se o cilindro estiver completo, então a ação *atirar* produz o efeito determinístico de matar o peru. Se o cilindro estiver totalmente vazio, então *atirar* não mata o peru. Qualquer outra situação, ou seja, a arma está carregada mas não se sabe se a bala está na agulha ou não, pode ou não matar o peru. Tal cenário é representado em $\mathcal{LAP}_{\rightsquigarrow}$ da seguinte maneira:

$$LEIS = \left\{ \begin{array}{l} \Box\langle esperar\rangle\top, \\ \Box\langle carregar\rangle\top, \\ \Box\langle atirar\rangle\top, \\ \Box(Cheio \rightarrow Carregada), \\ \Box[carregar]Carregada, \\ \Box[atirar]\neg Cheio, \\ \Box(Cheio \rightarrow [atirar]\neg Vivo) \end{array} \right\}$$

$$\rightsquigarrow\ = \left\{ \begin{array}{l} carregar \rightsquigarrow Cheio, \\ atirar \rightsquigarrow \neg Cheio, \\ atirar \rightsquigarrow \neg Vivo \end{array} \right\}$$

Este cenário, tal qual apresentado, já permite a dedução das conclusões esperadas, não havendo, portanto, a necessidade de inclusão de axioma de persistência algum. Entretanto, a utilização pura e simples da hipótese de fecho de explicação tal como definida em [56] levaria à obtenção do seguinte axioma de persistência condicional:

$$\Box((\neg Cheio \wedge Vivo) \rightarrow [atirar]Vivo),$$

o qual seria conseqüência de algum axioma de fecho de explicação que, erroneamente, expressaria que *Vivo* se torna falso *somente se Cheio* é verdadeiro. Como na abordagem aqui analisada a hipótese de fecho de explicação não está embutida na lógica de base, tal axioma jamais seria acrescentado à teoria. ∎

Mas então resta uma pergunta: como saber quando se deve ou não adicionar axiomas de persistência condicionais à representação do domínio? A resposta é dada em [4, seção 7.2], onde é apresentado um algoritmo interativo para obtenção dos axiomas de persistência condicionais cuja presença na teoria pode ser considerada razoável.

Da mesma forma, em [4, seção 7.1] é apresentado um algoritmo para gerar semi-automaticamente a relação de dependência, enquanto que em [4, seção 7.3] é proposto um algoritmo para obtenção das leis de executabilidade, de uma maneira similar à compleção de Clark [11], numa tentativa de sobrepujar os problemas relatados no exemplo 6.4.2.

Uma questão interessante em relação à abordagem aqui apresentada é se ela realmente conta como uma solução para o problema da ramificação, uma vez que com ela, apesar de não precisarmos enunciar na lógica de base leis envolvendo efeitos indiretos, necessitamos explicitar as dependências indiretas do domínio.

Um argumento favorável a esse método é o fato de que, procedendo dessa maneira, garante-se a aplicabilidade do formalismo a uma gama muito maior de problemas, enquanto que as demais abordagens presentes na literatura são restritas a apenas algumas classes de teorias de ações. Um exemplo bastante notório ilustrando tal argumento é o cenário da porta bloqueada, representado no exemplo 6.4.2, o qual traz uma série de problemas aos demais métodos, conforme vimos ao longo deste trabalho.

Além disso, o método aqui analisado não é o único a enunciar dependências causais indiretas. Observando atentamente, percebe-se que o passo 3 da minimização da causalidade, analisada na seção 3.3 e mais claramente visível no exemplo 3.4.1, determina que se enunciem todas as regras causais possíveis para o domínio, o que em essência é a mesma tarefa empreendida com o uso de relações de dependência. Da mesma forma, conforme vimos no exemplo 5.4.2, o axioma de fecho de explicação (5.17) também codifica explicitamente na lógica de base um efeito indireto.

Entretanto, a diferença fundamental a ser notada aqui é o fato de as dependências indiretas serem enunciadas no nível da metalinguagem, e não no da linguagem-objeto, como nas propostas supracitadas. Essa diferença, apesar de sutil, é realmente importante ao se considerar a flexibilidade do formalismo. Nesse sentido, com a abordagem aqui investigada, uma vez representado o domínio, mantém-se a lógica de base inalterada, promovendo-se alterações eventualmente necessárias apenas na relação de dependência.

Em termos de fundamentos, a abordagem analisada neste capítulo pode, de fato, ser considerada como uma abordagem monotônica. Entretanto, o processo de raciocínio definido utiliza uma espécie de negação por falha, baseada na não pertinência ao conjunto que denota a dependência, o que faz com que, dependendo do ponto de vista, a sua relação de inferência possa ser enxergada como não-monotônica, com respeito a dependências [7].

Antes de finalizarmos esta seção, queremos tocar num ponto fundamental da presente abordagem. No trabalho realizado em [7], supõe-se que todos os axiomas de persistência se referem a literais, isto é, enunciam a não-mudança de literais, ao invés da de fórmulas complexas da lógica. A razão para tal suposição, conforme justificado, se deve à dificuldade de encontrar exemplos nos quais axiomas de persistência envolvendo conjunções ou disjunções apareçam de maneira natural.

Não queremos abrir um debate aqui, mas afirmamos que pelo menos axiomas de persistência sobre conjunções aparecem de forma bastante natural. Um exemplo típico de uma situação desse tipo é a própria representação do cenário do tiro em Yale, ilustrado no exemplo 6.4.1. Nessa modelagem, percebe-se que o axioma de persistência condicional

$$\Box((\neg Carregada \wedge Vivo) \rightarrow [atirar]Vivo)$$

é uma conseqüência lógica do seguinte axioma de persistência puro sobre uma fórmula:

$$\Box((\neg Carregada \wedge Vivo) \rightarrow [atirar](\neg Carregada \wedge Vivo)). \tag{6.6}$$

Conjecturamos que situações desse tipo ocorrem em casos em que os literais componentes da fórmula complexa não estão relacionados por meio de uma restrição de domínio. Note que, para esse exemplo, $\neg Carregada$ e $Vivo$ são completamente independentes.

Por outro lado, se para o mesmo cenário tivéssemos ainda a restrição de domínio

$$\Box(Respirando \rightarrow Vivo), \tag{6.7}$$

então o axioma de persistência sobre fórmulas

$$\Box((Respirando \wedge Vivo) \rightarrow [carregar](Respirando \wedge Vivo)), \tag{6.8}$$

apesar de perfeitamente intuitivo, não é natural, uma vez que contém informação redundante. Neste caso, bastaria o axioma de persistência sobre o literal $Respirando$

$$\Box(Respirando \rightarrow [carregar]Respirando), \tag{6.9}$$

o qual, combinado com a restrição de domínio (6.7), acarreta o também axioma de persistência sobre literal a seguir:

$$\Box(Vivo \rightarrow [carregar]Vivo). \tag{6.10}$$

Observando mais atentamente, nota-se ainda a seguinte propriedade: veja que o axioma de persistência (não-natural) sobre fórmulas (6.8) pode ser representado pelos dois axiomas de persistência sobre literais (6.9) e (6.10). Já o axioma de persistência sobre fórmulas (6.6), o qual é perfeitamente natural, não é equivalente aos seguintes axiomas de persistência sobre literais dele dedutíveis:

$$\Box(\neg Carregada \rightarrow [atirar]\neg Carregada) \tag{6.11}$$

$$\Box(Vivo \rightarrow [atirar]Vivo) \tag{6.12}$$

Isso ocorre porque (6.12) não constitui um axioma de persistência válido no domínio em consideração. Além do mais, tal axioma tornaria a teoria inconsistente.

Nesse sentido, tais fatos apóiam a necessidade de se ver com outros olhos conjunções de literais em função de seu relacionamento via restrições de domínio.

Sendo assim, como conclusão do raciocínio acima desenvolvido, é razoável dizer que, para os exemplos considerados:

- Se dois ou mais literais estão relacionados por uma restrição de domínio, então o potencial axioma de persistência sobre sua conjunção não surge de maneira natural, pois pode ser substituído por vários axiomas de persistência sobre literais, cada qual representando a não-mudança de um dos literais da conjunção;

- Se, do contrário, os literais em questão forem independentes, então o axioma de persistência que denota a não-mudança de sua conjunção surge de maneira natural.

Seja como for, tais propriedades mereceriam antes uma investigação mais rigorosa a fim de poderem ser formalizadas. Conjectura-se ainda que tal formalização passaria antes por uma definição precisa do *contexto* no qual as ações são executadas. Essa constitui uma das principais motivações para a próxima parte deste trabalho.

Parte III

Uma Nova Proposta Causal

Capítulo 7

Dependência contextual

Vivemos numa sociedade de dependências que se expandem em progressão geométrica.

— José Saramago

Como tratar os problemas da persistência e da ramificação e ainda poder raciocinar sobre ações não-determinísticas sem a necessidade de enunciar axiomas de persistência condicionais? Uma proposta no sentido de atingir esse objetivo é apresentada neste capítulo. Apresentaremos aqui a lógica $\mathcal{LAPD}$, que consiste de $\mathcal{LAP}$ acrescida de uma nova relação de dependência, capaz de capturar os *contextos* nos quais as ações são executadas.

Basicamente, empreenderemos aqui a definição de uma relação de dependência envolvendo *ações* e *literais* com o acréscimo de um parâmetro capaz de caracterizar o *contexto* (situação particular) em que as referidas ações podem influenciar os literais em questão. Conforme ficará evidente no decorrer do capítulo, a definição de uma dependência em termos de contextos estabelece uma noção causal mais informativa, eliminando a necessidade de se enunciar axiomas de persistência condicionais na descrição de domínio.

7.1 Considerações e definição

Antes de estabelecermos a formalização da nova noção causal proposta, convém fazer algumas considerações a seu respeito.

Essencialmente, a relação de dependência contextualizada consistirá de uma *relação ternária* entre ações, literais e fórmulas[1]. Neste caso, determinadas fórmulas serão consideradas como *contextos* nos quais uma ação é executada, tomando parte no processo de produção do resultado final esperado.

Sendo assim, dizer que um literal L *depende* de uma determinada ação α em um dado *contexto* C significará, a partir de então, dizer que se C é verdadeiro, então, após a

[1]Fórmulas da lógica clássica proposicional, sem os operadores modais.

execução de α, L *pode* ter o seu valor de verdade alterado.

Esse conceito será representado pela expressão α $\mathsf{may\ cause}$ L if C, a qual constitui um elemento da nova relação que estamos definindo e é lida como "α pode causar L no contexto C".

Como exemplo, considere a ação *atirar* e os literais $\neg\,Vivo$ e *Carregada*. Como *atirar* pode causar $\neg\,Vivo$ numa circunstância em que se tenha *Carregada*, então a expressão

$$atirar\ \mathsf{may\ cause}\ \neg\,Vivo\ \mathsf{if}\ Carregada$$

pertence à relação de dependência contextualizada.

Um ponto importante a ser destacado é o fato de a noção de dependência aqui introduzida apenas *permitir* mudança, e não necessariamente a *forçar*.

Utilizamos $A\c{C}\tilde{O}ES$ e LIT tais quais definidos no capítulo 6. O conjunto de todas as fórmulas da lógica clássica proposicional será denotado por $FORPROP$.

Definição 7.1.1 Uma *relação de dependência contextual* é uma relação ternária $\mathcal{D} \subseteq A\c{C}\tilde{O}ES \times LIT \times FORPROP$. ∎

As triplas (α, L, C) serão escritas como expressões da forma α $\mathsf{may\ cause}$ L if C e representam o fato de que "L *pode* ser causado por α no contexto C". Em outras palavras, isso significa que "a execução da ação α *pode* alterar o valor de verdade do literal L, desde que a fórmula C seja verdadeira".

Observação 7.1.1 Contextos como disjunção aparentam ser raros. Sendo assim, conjecturamos que, na prática, a fórmula denotando o contexto C em geral será uma *conjunção* de literais. Seja como for, a expressão α $\mathsf{may\ cause}$ L if $C_1 \vee C_2$ pode ser substituída por α $\mathsf{may\ cause}$ L if C_1 e α $\mathsf{may\ cause}$ L if C_2. Portanto, podemos supor, sem perda de generalidade, que contextos são conjunções de literais.

7.2 Uma nova lógica de ações e planos

De maneira análoga à empreendida na abordagem com a dependência binária $\rightsquigarrow$, vista no capítulo 6, podemos combinar a lógica de ações e planos $\mathcal{LAP}$ com a nova noção de dependência causal contextualizada $\mathcal{D}$, obtendo, assim, a lógica $\mathcal{LAPD}$. Nesse sentido, os $\mathcal{LAP}$-modelos deverão satisfazer a condição de que sempre que todos os contextos nos quais uma ação pode causar um dado literal L forem falsos, então o valor de verdade de L deverá ser preservado após a execução de tal ação.

Como exemplo, considere a ação *atirar* e o literal $\neg\,Vivo$ no cenário do tiro em Yale. Neste caso, temos que a única maneira de *atirar* causar $\neg\,Vivo$ é quando *Carregada* for verdadeiro. Assim, numa circunstância em que se tenha $\neg Carregada$, a persistência de $Vivo$ será garantida pela falsidade do contexto $Carregada$.

Para o mesmo cenário, considerando a ação $esperar$ e o literal $Carregada$, como $esperar$ não pode causar $\neg Carregada$, não haverá em $\mathcal{D}$ nenhuma expressão da forma $esperar$ may cause $\neg Carregada$ if C, qualquer que seja o contexto C. Sendo assim, garante-se a permanência de $Carregada$ após $esperar$.

Com essa condição de dependência, os $\mathcal{LAP}$-modelos nos quais mudanças não intuitivas ocorrem são eliminados da seguinte maneira: suponhamos que estamos numa situação particular (um mundo possível) w onde o literal L é falso. Primeiro, imagine que o único elemento de $\mathcal{D}$ envolvendo ambos α e L seja α may cause L if C. Então, se C for verdadeiro, a execução de α pode causar, ou não, uma mudança no valor de verdade de L. Por outro lado, certamente α não alterará o valor de verdade de L se C for falso. Suponhamos agora que não existe $C \in PFOR$ tal que α may cause L if $C \in \mathcal{D}$. Então, a execução de α jamais tornará L verdadeiro, e assim L continuará sendo falso após se executar α.

Definição 7.2.1 Seja $\mathcal{D}$ uma relação de dependência ternária. Um modelo de $\mathcal{LAPD}$ é um $\mathcal{LAP}$-modelo $\mu = \langle W, \{R_\alpha : \alpha \in A\c{C}\tilde{O}ES\}, R_\Box, \tau\rangle$, tal que para $\alpha \in A\c{C}\tilde{O}ES$, $L \in LIT$ e $w, w' \in W$, sempre que $wR_\alpha w'$, se não existe $C \in FORPROP$ tal que $w \models C$ e α may cause L if C, então $w \notin \tau(L)$ se e somente se $w' \notin \tau(L)$. ■

Observação 7.2.1 No caso em que para algum $\alpha \in A\c{C}\tilde{O}ES$ e um dado $L \in LIT$ não há nenhuma expressão α may cause L if C em $\mathcal{D}$, qualquer que seja $C \in FORPROP$, essa definição também aplica, conforme esperado. A explicação é dada pelo fato de, neste caso, o antecedente da definição 7.2.1 ser verdadeiro.

A figura 7.1 ilustra a semântica de $\mathcal{LAPD}$ dada na definição 7.2.1. 7.1–(a) e 7.1–(b) representam duas circunstâncias em que se tem α may cause L if C na dependência. Em 7.1–(a), devido à validade do contexto C, L pode se tornar verdadeiro ou continuar falso. Em 7.1–(b), o valor de $\neg L$ é preservado após a execução de α devido à falsidade de C. 7.1–(c) representa um caso em que uma dada ação α nunca afeta um dado literal $\neg L$, preservando-o após sua execução.

Dada uma relação de dependência $\mathcal{D}$, uma fórmula A é verdadeira em um $\mathcal{LAPD}$-modelo $\mu = \langle W, \{R_\alpha : \alpha \in A\c{C}\tilde{O}ES\}, R_\Box, \tau\rangle$ se $w \models A$ para todo $w \in W$. A é $\mathcal{LAPD}$-válida (denotado $\models_{\mathcal{LAPD}} A$) se A for verdadeira em todos os $\mathcal{LAPD}$-modelos.

Observação 7.2.2 No caso em que $\mathcal{D} = A\c{C}\tilde{O}ES \times LIT \times FORPROP$, o conjunto dos $\mathcal{LAPD}$-modelos é exatamente o conjunto dos $\mathcal{LAP}$-modelos. Por outro lado, quando $\mathcal{D} = \emptyset$, as ações do domínio representado não possuem efeito algum e os modelos podem ser restringidos como possuindo apenas um único mundo possível.

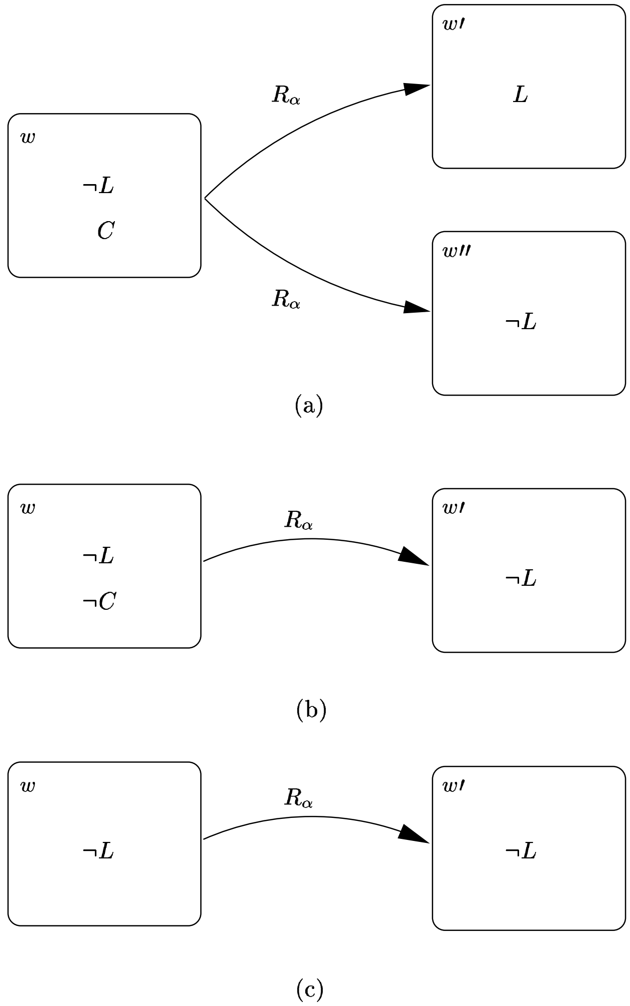

Figura 7.1: Ilustração de transição entre mundos considerando a semântica de $\mathcal{LAPD}$ dada na definição 7.2.1.

7.3 Inferindo axiomas de persistência condicionais

Uma vez definida a dependência em função de contextos, automaticamente temos uma solução para os axiomas de persistência condicionais. Isso será demonstrado no decorrer da presente seção.

Definição 7.3.1 Seja $\alpha \in AÇÕES$, $L \in LIT$ e $\mathcal{D}$ uma relação de dependência. Definimos

$$Pre_{\mathcal{D}}(\alpha, L) = \bigvee_{(\alpha \textsf{ may cause } L \textsf{ if } C) \in \mathcal{D}} C$$

■

Em outras palavras, $Pre_{\mathcal{D}}(\alpha, L)$ é a disjunção de todos os contextos nos quais α pode causar uma mudança no valor de verdade do literal L, dada uma relação de dependência $\mathcal{D}$. Com isso, temos o seguinte teorema fundamental:

Teorema 7.3.1 $\models_{\mathcal{LAPD}}\Box((\neg Pre_{\mathcal{D}}(\alpha, L) \wedge \neg L) \rightarrow [\alpha]\neg L)$.

Prova: Suponha que $\Box((\neg Pre_{\mathcal{D}}(\alpha, L) \wedge \neg L) \rightarrow [\alpha]\neg L)$ seja falso, isto é, existe um mundo possível w tal que $w \models \neg Pre_{\mathcal{D}}(\alpha, L)$ e $w \models \neg L$, e não é o caso que $w \models [\alpha]\neg L$, ou seja, $w \models \langle\alpha\rangle L$. Suponha que α é executável, ao menos quando $\neg Pre_{\mathcal{D}}(\alpha, L)$ e $\neg L$ são verdadeiros em w. Então existe um mundo possível w' tal que $wR_\alpha w'$ e $w' \models L$. Como $w \models \neg Pre_{\mathcal{D}}(\alpha, L)$, temos que para toda expressão α may cause L if C em $\mathcal{D}$, $w \not\models C$, e como $w \models \neg L$, pela definição de $\mathcal{D}$, devemos ter $w' \models \neg L$, o que é um absurdo. ∎

Com esse resultado, temos que numa descrição de domínio usando uma relação de dependência $\mathcal{D}$ não há a necessidade de enunciar um conjunto de axiomas de persistência condicionais, uma vez que todas as conclusões que são obtidas com o auxílio destes últimos podem também ser inferidas por meio de $\mathcal{D}$.

Como um exemplo, considere a ação *atirar* e os literais *Vivo* e $\neg$*Carregada*, e suponha

$$\mathcal{D} = \{atirar \text{ may cause } \neg Vivo \text{ if } Carregada\}$$

Então o axioma de persistência condicional

$$\Box((\neg Carregada \wedge Vivo) \rightarrow [atirar]Vivo) \tag{7.1}$$

é $\mathcal{LAPD}$-válido. Em outros termos, a persistência de *Vivo* quando $\neg$*Carregada* é verdadeiro segue da informação de dependência em $\mathcal{D}$, tornando completamente desnecessário, assim, o enunciado do axioma de persistência condicional (7.1).

Na seqüência, definimos uma axiomática para $\mathcal{LAPD}$ capaz de representar a noção causal da dependência contextualizada $\mathcal{D}$ junto aos axiomas definidos para $\mathcal{LAP}$.

7.4 Axiomática de $\mathcal{LAPD}$

Dada uma relação de dependência ternária $\mathcal{D}$, axiomatizamos a classe dos $\mathcal{LAPD}$-modelos da mesma forma como feito para $\mathcal{LAP}$ [4], acrescentando um esquema de axioma baseado na condição de dependência:

- $Persist([\alpha]) : \neg L \rightarrow [\alpha]\neg L$, se $\neg Pre_{\mathcal{D}}(\alpha, L)$.

A fim de mostrar a adequação e a completude da axiomática aqui definida para $\mathcal{LAPD}$ em relação à sua semântica, temos o seguinte teorema:

Teorema 7.4.1 Para toda relação de dependência ternária $\mathcal{D}$ sobre *AÇÕES* $\times$ *LIT* $\times$ *FORPROP*, a axiomática de $\mathcal{LAPD}$ é adequada e completa com respeito à classe dos $\mathcal{LAPD}$-modelos.

Prova: Veja apêndice A. ∎

Na próxima seção, veremos que o acréscimo da relação de dependência $\mathcal{D}$ à lógica de base não constitui um aumento na sua complexidade.

7.5 Complexidade de $\mathcal{LAPD}$

O seguinte teorema estabelece que a complexidade de $\mathcal{LAPD}$ é a mesma de $\mathcal{LAP}$.

Teorema 7.5.1 Em $\mathcal{LAPD}$, o problema da satisfatibilidade é EXPTIME-completo.
Prova: $\mathcal{LAP}$ é tão somente $\mathcal{LAPD}$ com $\mathcal{D} = A\c{C}\tilde{O}ES \times LIT \times FORPROP$. Com isso, a complexidade de $\mathcal{LAPD}$ é no mínimo EXPTIME. Para estabelecer o limite superior, basta definir uma transformação polinomial t de $\mathcal{LAPD}$ em $\mathcal{LAP}$.

Seja A uma fórmula de $\mathcal{LAPD}$ e seja a transformação

$$t(A) = A \wedge \bigwedge_{\alpha \text{ e } L \text{ aparecem em } A} (\neg Pre_{\mathcal{D}}(\alpha, L) \wedge \neg L) \rightarrow [\alpha]\neg L$$

Note que $Pre_{\mathcal{D}}(\alpha, L)$ é finito e que sua cardinalidade é quadrática no tamanho de A. Sendo assim, temos que $t(A)$ é uma transformação polinomial e a complexidade computacional de $\mathcal{LAPD}$ é no máximo EXPTIME, donde segue o resultado. ■

Na seqüência, empreenderemos a representação de alguns cenários em $\mathcal{LAPD}$, a fim de ilustrar as principais vantagens obtidas com seu formalismo.

7.6 Cenários-exemplos em $\mathcal{LAPD}$

Nesta seção, apresentamos alguns cenários-exemplos de utilização de $\mathcal{LAPD}$ como formalismo de base para modelagem e realização de inferências sobre ações. Perceba que em nenhum cenário há a necessidade de se acrescentar axiomas de persistência condicionais, dado que o aspecto semântico dos mesmos é capturado de maneira implícita por meio da relação de dependência contextual $\mathcal{D}$.

Exemplo 7.6.1 (Seqüência do exemplo 6.4.1) A representação do cenário do tiro em Yale em $\mathcal{LAPD}$ é a seguinte:

$$LEIS = \left\{ \begin{array}{l} \Box\langle esperar\rangle\top, \\ \Box\langle carregar\rangle\top, \\ \Box\langle atirar\rangle\top, \\ \Box[carregar]Carregada, \\ \Box[atirar]\neg Carregada, \\ \Box(Carregada \rightarrow [atirar]\neg Vivo) \end{array} \right\}$$

$$\mathcal{D} = \left\{ \begin{array}{c} carregar \ \textsf{may cause} \ Carregada \ \textsf{if} \ \top, \\ atirar \ \textsf{may cause} \ \neg Vivo \ \textsf{if} \ Carregada, \\ atirar \ \textsf{may cause} \ \neg Carregada \ \textsf{if} \ \top \end{array} \right\}$$

$$BC = \{\neg Carregada, Vivo\}$$

Da mesma forma que no exemplo 6.4.1, a partir dessa modelagem, é possível inferir:

$$\models_{\mathcal{LAPD}} (BC \wedge LEIS) \rightarrow [carregar][esperar][atirar](\neg Carregada \wedge \neg Vivo)$$

A figura 7.2 a seguir ilustra as transições entre os mundos possíveis deste cenário.

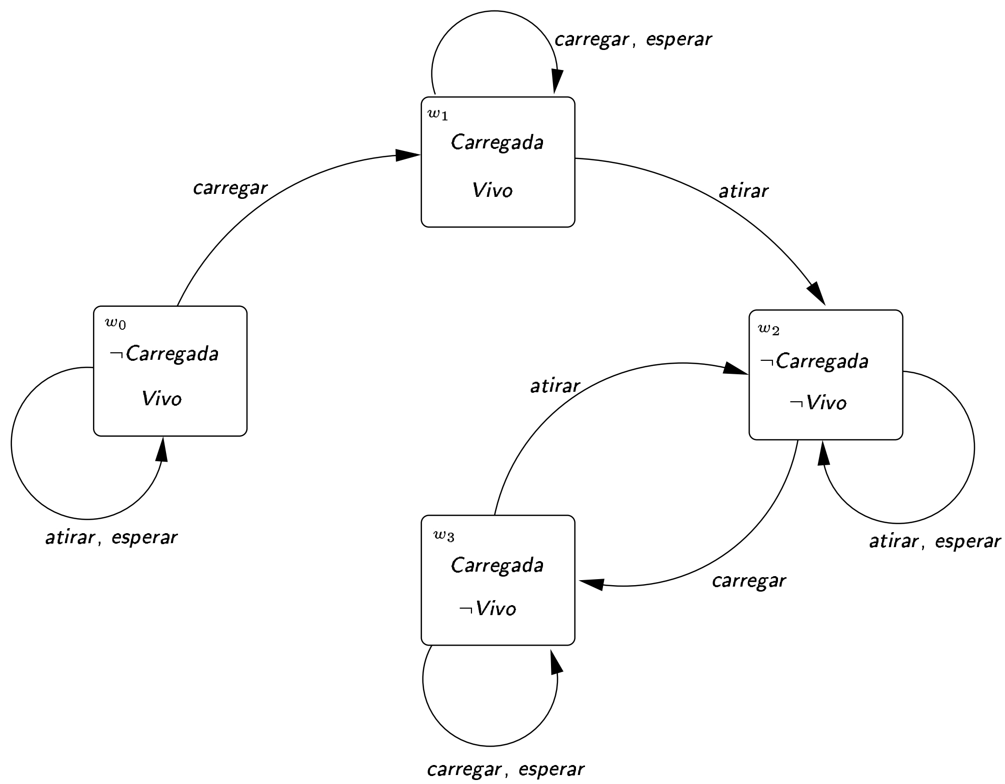

Figura 7.2: Transições entre os mundos possíveis do cenário do tiro em Yale.

Note que, para esse exemplo, a representação de *LEIS*, ao contrário da do exemplo 6.4.1, não necessita de axiomas de persistência condicionais. Dessa forma, a representação da relação de dependência $\mathcal{D}$ é mais intuitiva do que a apresentada com a dependência binária no exemplo 6.4.1. ■

No próximo exemplo, utilizamos $\mathcal{LAPD}$ na representação de um cenário envolvendo ramificações e qualificações implícitas.

Exemplo 7.6.2 (Seqüência do exemplo 6.4.2) O cenário da porta bloqueada é representado em $\mathcal{LAPD}$ da seguinte maneira:

$$LEIS = \left\{ \begin{array}{l} \Box(Bloqueada \rightarrow Fechada), \\ \Box[abrir]\neg Fechada, \\ \Box[arrombar]\neg Fechada \end{array} \right\}$$

$$\mathcal{D} = \left\{ \begin{array}{c} abrir \text{ may cause } \neg Fechada \text{ if } \top, \\ arrombar \text{ may cause } \neg Fechada \text{ if } \top, \\ arrombar \text{ may cause } \neg Bloqueada \text{ if } \top \end{array} \right\}$$

É importante ressaltar que as leis de executabilidade não foram enunciadas acima justamente porque se as tivéssemos inserido, tal cenário ficaria inconsistente. Na verdade, leis de executabilidade só são necessárias para a realização de tarefas de planificação.

Note que, tal como no exemplo 6.4.2, também necessitamos escrever a dependência indireta entre *arrombar* e *Bloqueada*. Entretanto, valem aqui os mesmos comentários tecidos na seção 6.5.

A figura 7.3 ilustra as transições entre os mundos possíveis deste cenário.

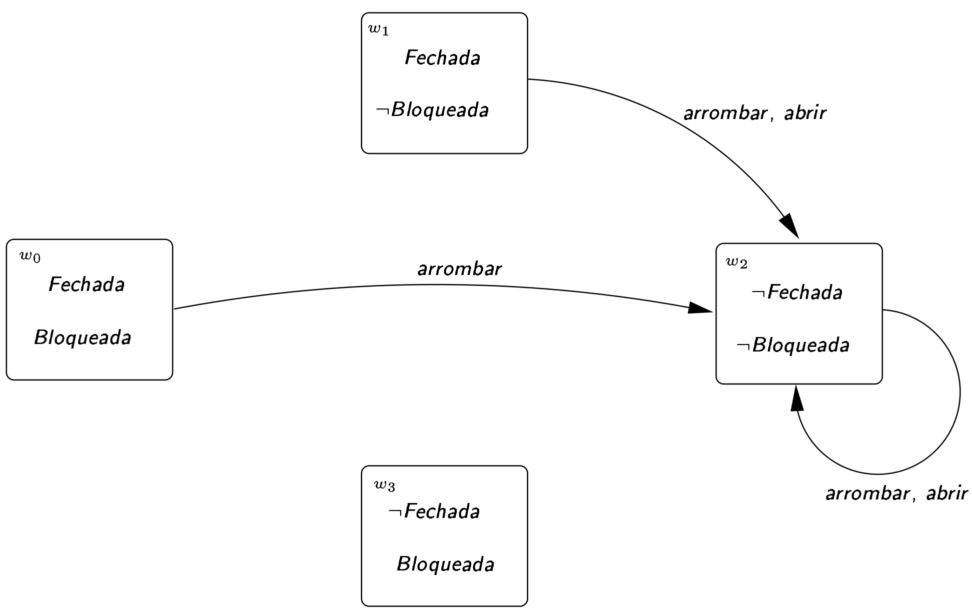

Figura 7.3: Transições entre os mundos possíveis do cenário da porta bloqueada.

A partir da modelagem apresentada e considerando as observações

$$BC = \{Fechada, Bloqueada\},$$

é possível mostrar que

$$\models_{\mathcal{LAPD}}(BC \land LEIS) \rightarrow [arrombar](\neg Fechada \land \neg Bloqueada),$$

o que significa que em $\mathcal{LAPD}$, assim como em $\mathcal{LAP}_{\leadsto}$, podemos derivar o efeito indireto da ação *arrombar*.

Com essa descrição de domínio, é possível ainda mostrar que

$$\models_{\mathcal{LAPD}}(BC \land LEIS) \rightarrow [abrir]\bot,$$

representando o fato de que com $\mathcal{LAPD}$ pode-se derivar também a qualificação implícita da ação *abrir*, isto é, tal ação não pode ser executada se a porta estiver bloqueada. ∎

Esses exemplos ilustram algumas das vantagens da definição de uma dependência causal contextualizada em $\mathcal{LAP}$. Entretanto, para que tais vantagens possam ser sentidas em termos práticos, resta ainda a definição de um método de prova adequado e completo para a realização de inferências em $\mathcal{LAPD}$. É justamente esse o tema do próximo capítulo.

Capítulo 8

Método de prova

Qualquer idéia poderosa é absolutamente fascinante e inútil até decidirmos usá-la.

— Richard Bach

Neste capítulo, apresentamos a definição de um método de prova para a realização de inferências na lógica de base $\mathcal{LAPD}$, fundamentado nos *sistemas de tableau semânticos* [18, 25]. Em essência, o método aqui definido é uma modificação do proposto para $\mathcal{LAP}_{\leadsto}$ [4], com as adaptações necessárias a comportar a noção de dependência ternária definida no capítulo 7. Resultados importantes como adequação, completude e decidibilidade são também demonstrados.

8.1 Sistema de tableau para $\mathcal{LAPD}$

O método de tableau que definiremos para $\mathcal{LAPD}$ é baseado nos sistemas de tableau modais com relação de acessibilidade explícita [25]. Em essência, o método aqui elaborado é bastante similar ao definido em [4] para o caso de $\mathcal{LAP}$ com a dependência binária $\leadsto$.

Numa primeira instância, a única coisa a ser feita é adicionar ao método de tableau para $\mathcal{LAP}$ [4] uma maneira de se captar a noção semântica da condição de dependência contextualizada:

- Toda vez em que ocorrer a execução de uma dada ação α em um mundo possível w, sendo o literal L falso em w, se, para todo contexto $C \in FORPROP$ tal que α $\mathsf{may\ cause}$ L if C, $w \not\models C$, então devemos *propagar* o literal $\neg L$ à situação resultante da execução de α.

Entretanto, essa simples modificação não é suficiente. Considerando, por exemplo, uma situação em que se tenha α $\mathsf{may\ cause}$ L if C, com $w \not\models C$, e β $\mathsf{may\ cause}$ $\neg L$ if C, com $w \not\models C$, para $L \in LIT$, C um contexto nos moldes do capítulo 7 e $\alpha, \beta \in AÇÕES$, teremos que a fórmula $\langle\alpha\rangle\neg L \wedge \langle\beta\rangle L$, que é $\mathcal{LAPD}$-insatisfatível, não é provada como tal pelo método proposto. Esse problema é resolvido através da propagação de certos literais a estados anteriores do processo de prova, de maneira similar à empreendida em [4].

Sendo assim, adiciona-se também à definição do $\mathcal{LAP}$-tableau uma regra que permita propagar um determinado literal à situação que originou o estado atual da prova. Conforme ficará mais evidente ao longo do presente capítulo, a utilização desse procedimento permite a propagação de literais não causados por uma dada ação em direção à raiz da prova, o que, apesar de não ser comum na literatura, oferece uma nova maneira de se enxergar os métodos de prova por tableau [4].

Na seqüência, apresentamos a definição do método de tableau para $\mathcal{LAPD}$ propriamente dito, estabelecendo antes, porém, a terminologia utilizada.

Definição 8.1.1 Uma *fórmula rotulada* é uma estrutura da forma $w :: A$, onde w é um mundo possível de um modelo de Kripke e $A \in FOR$. ■

Uma fórmula rotulada $w :: A$ representa o fato de que a fórmula A é verdadeira no mundo possível w.

Definição 8.1.2 Uma *estrutura de transição*[1] Υ é uma relação ternária $\Upsilon \subseteq (AÇÕES \cup \Box) \times W \times W$, onde W é um conjunto de mundos possíveis de um modelo de Kripke. Os elementos (α, w, w') de Υ são denotados $w \stackrel{\alpha}{\to} w'$ e representam o fato de que o mundo w' é acessível ao mundo w através de uma transição por (execução de) α. ■

Definição 8.1.3 Uma *árvore*[2] é um par $\langle S, \Upsilon\rangle$, onde S é um conjunto de fórmulas rotuladas e Υ é uma estrutura de transição. ■

Definição 8.1.4 Uma *regra de tableau* ρ é uma regra da forma:

$$\rho\ \frac{\mathcal{N};\Gamma}{\mathcal{D}_1|\mathcal{D}_2|\ldots|\mathcal{D}_k;\Gamma'}$$

onde ρ é o *nome da regra*, $\mathcal{N};\Gamma$ é o *numerador* e $\mathcal{D}_1|\mathcal{D}_2|\ldots|\mathcal{D}_k;\Gamma'$ é o *denominador*. ■

No numerador de uma regra ρ, $\mathcal{N}$ representa uma ou mais fórmulas rotuladas $w :: A$, chamadas de *fórmulas principais* da regra, separadas por ",". Γ representa uma *condição adicional* a ser satisfeita para a aplicação da regra. Se não houver tal condição, Γ é omitida.

[1] A idéia intuitiva por detrás de uma estrutura de transição é representar a relação de acessibilidade, da mesma forma como utilizado em [53].

[2] A idéia de árvore, tal qual definida em termos de conjunto de fórmulas relacionadas por meio de uma estrutura de transição, pode ser enxergada como um modelo de Kripke.

No denominador, cada $\mathcal{D}_i, 1 \leq i \leq k$, possui uma ou mais fórmulas rotuladas, chamadas de *fórmulas laterais* da regra de tableau. O símbolo "|" representa a ocorrência de uma *ramificação* na estrutura de transição. Γ' é uma *condição adicional* a ser satisfeita após a aplicação da regra. Se não houver tal condição, omite-se Γ'.

O nome da regra ρ usualmente consiste do conectivo mais externo da fórmula principal, entre parênteses, podendo também consistir de um nome mais complexo.

A figura 8.1 apresenta as regras de tableau definidas para $\mathcal{LAPD}$.

Regras de Tableau:

$$(\bot)\frac{w :: A, w :: \neg A}{w :: \bot} \qquad (\neg)\frac{w :: \neg\neg A}{w :: A} \qquad (\wedge)\frac{w :: A \wedge B}{w :: A, w :: B}$$

$$(\vee)\frac{w :: \neg(A \wedge B)}{w :: \neg A | w :: \neg B} \qquad (K_\alpha)\frac{w :: [\alpha]A; w \overset{\alpha}{\to} w'}{w' :: A} \qquad (T_\Box)\frac{w :: \Box A}{w :: A}$$

$$(4_\Box)\frac{w :: \Box A; w \overset{\Box}{\to} w' \text{ ou } \exists \alpha w \overset{\alpha}{\to} w'}{w' :: \Box A}$$

$$(\langle\alpha\rangle)\frac{w :: \neg[\alpha]A}{w' :: \neg A; w \overset{\alpha}{\to} w' \text{ para } w' \text{ novo}} \qquad (\Diamond)\frac{w :: \neg\Box A}{w' :: \neg A; w \overset{\Box}{\to} w' \text{ para } w' \text{ novo}}$$

$$(RP)\frac{w :: \neg L; w \overset{\alpha}{\to} w' \text{ e } w \not\models Pre_{\mathcal{D}}(\alpha, L)}{w' :: \neg L}$$

$$(RR)\frac{w' :: L; w \overset{\alpha}{\to} w' \text{ e } w \not\models Pre_{\mathcal{D}}(\alpha, L)}{w :: L}$$

Figura 8.1: Regras do $\mathcal{LAPD}$-tableau. w, w' são mundos possíveis, $\alpha \in A\c{C}\tilde{O}ES$ e $L \in LIT$. $Pre_{\mathcal{D}}(\alpha, L)$ é tal como na definição 7.3.1.

A regra $(\bot)$ é o *teste de inconsistência*. Toda vez em que um dado mundo possível contiver simultaneamente uma fórmula e sua negação, conclui-se a contradição $\bot$.

A regra $(\neg)$ é a *regra da dupla negação*, a qual tem a função de eliminar redundâncias nas fórmulas negadas.

$(\wedge)$ é a *regra da conjunção*, também chamada de *regra-e*. Tal regra representa o fato de que toda vez em que se observe que uma determinada conjunção é verdadeira em um dado mundo possível, então cada um de seus componentes o deve ser.

A regra $(\vee)$ é a *regra da disjunção*, ou ainda *regra-ou*. A aplicação de tal regra representa uma prova por casos, na qual se uma dada disjunção é verdadeira, então um dos dois disjuntos (ou ambos) também deve(m) ser. Conforme ilustrado na figura 8.1, a aplicação da regra-ou ocasiona uma ramificação no processo de prova.

A regra (K_α) traduz a idéia segundo a qual se em um dado mundo tem-se uma fórmula da forma $[\alpha]A$, então em todo mundo que lhe é acessível por α deve-se ter A. Com essa regra, captura-se a axiomática definida pelo sistema de lógica multimodal K.

Com a regra $(T_\Box)$, podemos representar a propriedade reflexiva do sistema de lógica monomodal $S4$, segundo a qual se uma dada fórmula é necessária, então ela é verdadeira no próprio mundo atual.

A regra $(4_\Box)$, por sua vez, capta a transitividade da lógica monomodal $S4$.

Com as regras $(\langle\alpha\rangle)$ e $(\Diamond)$, estabelece-se que, se uma dada fórmula é possível, então ela deve ser verdadeira em algum mundo possível. Observe que a aplicação dessas regras, por ocasionar a criação de novos nodos no tableau durante o processo de prova, tem como conseqüência a modificação da estrutura de transição.

As duas últimas regras, (RP) e (RR), estão associadas à relação de dependência ternária definida no capítulo 7. A regra (RP), chamada de *regra de persistência*, formaliza a idéia de que se um dado literal em um nodo do tableau não teve seu valor de verdade alterado pela execução de uma dada ação α, então ele deve ser propagado ao nodo seguinte à execução de tal ação.

De maneira recíproca, a regra (RR), chamada de *regra de retro-propagação*, operacionaliza a idéia de que se um dado literal L é verdadeiro em um nodo mas não foi causado a mudar seu valor de verdade na última transição do tableau, então L também deve ser verdadeiro no nodo antecedente. Essa regra é necessária apenas para garantir a completude do método, conforme comentamos acima.

Observação 8.1.1 A aplicação das regras (RP) e (RR) requer a prova da validade ou não de $Pre_{\mathcal{D}}(\alpha, L)$, ou seja, de cada um dos contextos nos quais uma certa ação α pode causar um dado literal L. Neste sentido, a implementação de tais regras deverá levar em consideração a necessidade de realização de subprovas.

As regras $(\bot)$, $(\neg)$, $(\wedge)$, $(\vee)$ e $(T_\Box)$ são chamadas de *regras estáticas*, pois sua aplicação não provoca mudança do mundo possível atual. As regras (K_α), $(4_\Box)$, $(\langle\alpha\rangle)$ e $(\Diamond)$ são ditas *regras dinâmicas*, uma vez que sua aplicação resulta numa mudança de mundo possível. Já as regras (RP) e (RR) são chamadas de *regras de propagação*, cujo efeito é produzir alterações no conjunto de fórmulas de um dado nodo de maneira a melhor direcionar o procedimento de prova[3].

[3]Observe que, apesar de as regras (RP) e (RR) provocarem alterações em mundos diferentes do atual, não empreendem uma transição efetiva, tal como a realizada pelas regras dinâmicas.

Um aspecto interessante a ser observado com relação às regras da figura 8.1 é o fato de, em todas as regras, o mundo possível referido no denominador estar no máximo *um passo* distante do mundo representado no numerador da regra. Como exemplo, a regra $(T_\Box)$ adiciona a fórmula A no próprio mundo, $(\langle\alpha\rangle)$ adiciona $\neg A$ a um mundo sucessor, enquanto que (RR) adiciona L a um antecessor.

Definição 8.1.5 Um *sistema* (ou *cálculo*) *de tableau* $\mathcal{C}$L é uma coleção finita de regras de tableau, representada pelos nomes das regras, $\rho_1, \rho_2, \ldots, \rho_n$, onde L é a lógica de base considerada. ■

Nesse sentido, um cálculo de tableau para $\mathcal{LAPD}$ é dado por:

$$\mathcal{CLAPD} = \{(\perp), (\neg), (\wedge), (\vee), (K_\alpha), (T_\Box), (4_\Box), (\langle\alpha\rangle), (\Diamond), (RP), (RR)\}$$

Em função disso, temos que o método de tableau aqui definido para $\mathcal{LAPD}$ é fundamentado num cálculo de tableau com *regras de um passo*.

Definição 8.1.6 Uma regra de $\mathcal{LAPD}$-tableau ρ é *aplicável* a uma árvore $\langle S, \Upsilon\rangle$ se S contém uma instância das fórmulas principais de ρ e a condição Γ de ρ é satisfeita. ■

Definição 8.1.7 Um $\mathcal{LAPD}$*-tableau* para uma dada fórmula A é o limite de uma seqüência $\mathcal{F}^0, \mathcal{F}^1, \ldots$, de florestas (conjuntos de árvores) cujo estado inicial é $\mathcal{F}^0 = \{\langle w_0 :: A, \emptyset\rangle\}$ e cada $\mathcal{F}^{i+1}$ é obtido a partir de $\mathcal{F}^i$ pela aplicação de uma das regras de $\mathcal{CLAPD}$ a uma dada árvore $\langle S, \Upsilon\rangle \in \mathcal{F}^i$. ■

Sendo assim, um $\mathcal{LAPD}$-tableau pode ser visto como uma floresta particular, denotada por $\mathcal{F}^\infty$.

Definição 8.1.8 Uma árvore $\langle S, \Upsilon\rangle$ é *fechada* se $w_i :: \perp \in S$ para algum mundo possível w_i. Uma floresta é fechada se todas as suas árvores também o forem. Nesse sentido, um dado tableau é dito fechado se todas as suas árvores forem fechadas. ■

Finalizando as definições do método de tableau para $\mathcal{LAPD}$, temos ainda o seguinte conceito fundamental:

Definição 8.1.9 Uma *prova por refutação* para uma determinada fórmula A de $\mathcal{LAPD}$ através do método de tableau definido consiste na obtenção de um $\mathcal{LAPD}$-tableau fechado para $\neg A$. ■

Uma vez definido um método de prova por tableau para $\mathcal{LAPD}$, analisaremos, na próxima seção, um exemplo de sua aplicação a um cenário típico, com o objetivo de melhor ilustrar seu funcionamento.

8.2 Exemplo de prova

Apresentamos agora um exemplo de utilização do método de tableau para $\mathcal{LAPD}$ definido na seção anterior para a realização de inferências.

Relembrando o cenário do tiro em Yale [27], temos que a formalização do mesmo em $\mathcal{LAPD}$, tal qual apresentada na seção 7.6, é dada por:

$$LEIS = \left\{ \begin{array}{l} \Box\langle esperar\rangle\top, \\ \Box\langle carregar\rangle\top, \\ \Box\langle atirar\rangle\top, \\ \Box[carregar]Carregada, \\ \Box[atirar]\neg Carregada, \\ \Box(Carregada \rightarrow [atirar]\neg Vivo) \end{array} \right\}$$

A relação de dependência contextualizada é definida por:

$$\mathcal{D} = \left\{ \begin{array}{c} carregar \text{ may cause } Carregada \text{ if } \top, \\ atirar \text{ may cause } \neg Vivo \text{ if } Carregada, \\ atirar \text{ may cause } \neg Carregada \text{ if } \top \end{array} \right\}$$

Supondo uma configuração inicial do mundo representada por:

$$BC = \{\neg Carregada, Vivo\},$$

podemos concluir que

$$\models_{\mathcal{LAPD}}(BC \wedge LEIS) \rightarrow [carregar][esperar][atirar]\neg Vivo \tag{8.1}$$

A figura 8.2 ilustra uma prova por refutação para (8.1). A fim de facilitar a visualização, todas as informações comuns a todas as árvores do tableau foram fatorizadas. Na aplicação das regras de tableau, $w \stackrel{\alpha}{\rightarrow} w'$ significa "acrescentar (w, w') a Υ".

Observe na figura 8.2 como o literal *Carregada* é propagado até o mundo w_2 mediante a aplicação da regra (RP) na linha 8 do tableau.

Note também que as leis de executabilidade, por não serem necessárias para esse exemplo, foram omitidas da figura. Tais leis, entretanto, são fundamentais para tarefas de planificação.

A partir da próxima seção, provamos alguns resultados importantes a respeito do método de tableau definido para $\mathcal{LAPD}$.

1.	$w_0 :: (BC \wedge LEIS) \wedge \neg[carregar][esperar][atirar]\neg Vivo$		Início da prova: raiz do Tableau.
2a.	$w_0 :: \Box[carregar]Carregada$		$(\wedge)$ em **1.**
2b.	$w_0 :: \Box(Carregada \rightarrow [atirar]\neg Vivo)$		$(\wedge)$ em **1.**
2c.	$w_0 :: \neg[carregar][esperar][atirar]\neg Vivo$		$(\wedge)$ em **1.**
3.	$w_0 :: [carregar]Carregada$		$(T_\Box)$ em **2a.**
4.	$w_1 :: \neg[esperar][atirar]\neg Vivo$		$(\langle carregar\rangle)$ em **2c.**; $w_0 \overset{carregar}{\rightarrow} w_1$
5.	$w_2 :: \neg[atirar]\neg Vivo$		$(\langle esperar\rangle)$ em **4.**; $w_1 \overset{esperar}{\rightarrow} w_2$
6.	$w_3 :: \neg(\neg Vivo)$		$(\langle atirar\rangle)$ em **5.**; $w_2 \overset{atirar}{\rightarrow} w_3$
7.	$w_3 :: Vivo$		$(\neg)$ em **6.**
8.	$w_1 :: Carregada$		$(K_{carregar})$ em **3.**
9.	$w_2 :: \Box(Carregada \rightarrow [atirar]\neg Vivo)$		$(4_\Box)$ em **2b.**
10.	$w_2 :: Carregada \rightarrow [atirar]\neg Vivo$		$(T_\Box)$ em **9.**
11.	$w_2 :: \neg Carregada$	$w_2 :: [atirar]\neg Vivo$	$(\vee)$ em **10.**
12.	$w_2 :: Carregada$	$w_2 :: Carregada$	(RP) em **8.**
13.	$w_2 :: \bot$	$w_3 :: \neg Vivo$	$(\bot)$ em **11.** e **12.**, e (K_{atirar}) em **11.**
14.	fechada	$w_3 :: \bot$	$(\bot)$ em **7.** e **13.**
15.		fechada	Todas as árvores estão fechadas.

Figura 8.2: Uma demonstração por $\mathcal{LAPD}$-tableau para a fórmula $(BC \wedge LEIS) \rightarrow [carregar][esperar][atirar]\neg Vivo$.

8.3 Adequação do método

Na presente seção, provamos que o método de tableau para $\mathcal{LAPD}$ definido na seção 8.1 é adequado. Para tanto, precisaremos antes de uma definição e de um lema a ser provado.

Definição 8.3.1 Seja S um conjunto arbitrário de fórmulas rotuladas. Um *estado* $S(w)$ denota o conjunto de fórmulas A tais que $w :: A \in S$. ■

Em outros termos, $S(w)$ é o conjunto formado por todas as fórmulas verdadeiras no mundo possível w.

Lema 8.3.1 Sejam $\langle S, \Upsilon\rangle$ uma árvore finita e

$$\widehat{S(w)} = S(w) \wedge \bigwedge_{\substack{\alpha \in AÇÕES \\ w \xrightarrow{\alpha} w'}} \langle\alpha\rangle S(w') \wedge \bigwedge_{w \xrightarrow{\Box} w'} \Diamond S(w'),$$

onde $\langle\alpha\rangle S(w') = \{\langle\alpha\rangle A : A \in S(w')\}$ e $\Diamond S(w') = \{\Diamond A : A \in S(w')\}$.
Então, para toda regra, se para toda floresta $\mathcal{F}^{i+1} = \{\ldots, \langle S^{i+1}, \Upsilon^{i+1}\rangle, \ldots\}$ que pode ser obtida de $\mathcal{F}^i = \{\ldots, \langle S^i, \Upsilon^i\rangle, \ldots\}$ e para toda $\langle S^i, \Upsilon^i\rangle \in \mathcal{F}^i$ existe um w tal que $\widehat{S^{i+1}(w)}$ é insatisfatível, então $\mathcal{F}^i$ também é insatisfatível.

Prova: Prova-se esse resultado considerando cada uma das regras de tableau apresentadas na figura 8.1. Por questões de praticidade, consideraremos aqui apenas os casos envolvendo as regras $(\Diamond)$, $(\langle\alpha\rangle)$, (RP) e (RR). A fim de simplificar a notação, abreviaremos $\{(w, w') : w \xrightarrow{\alpha} w'\}$ por Υ_α.

- Regra $(\Diamond)$: S^i contém $w :: \Diamond\neg A$, assim, ao aplicar a regra $(\Diamond)$, adiciona-se $w \xrightarrow{\Box} w'$ a Υ^i e $w' :: \neg A$ a S^i. Se todos os $S^i(w)$ são satisfatíveis, então apenas $S^{i+1}(w')$ pode ser insatisfatível. Entretanto tem-se $\widehat{S^{i+1}(w')} = \neg A$. Supondo-se que $\neg A$ é insatisfatível, tem-se $\models_{\mathcal{LAPD}} A$, donde segue $\models_{\mathcal{LAPD}} \Box A$. Assim, $S^i(w)$ é insatisfatível também, uma vez que contém $\Diamond\neg A$.

- Regra $(\langle\alpha\rangle)$: análogo ao caso $(\Diamond)$.

- Regra (RP): Suponhamos que S^i contém $w :: \neg L$, com $L \in LIT$, Υ^i_α contém (w, w') e que $w \models \neg Pre_{\mathcal{D}}(\alpha, L)$. Ao aplicar a regra (RP), deve-se adicionar $w' :: \neg L$ a S^i. Suponhamos que $\widehat{S^{i+1}(w')} = \widehat{S^i(w') \cup \{\neg L\}}$ é insatisfatível. Então $\models_{\mathcal{LAPD}} \widehat{S^i(w')} \rightarrow L$. Entretanto, de $w :: \neg L$ e $w \models \neg Pre_{\mathcal{D}}(\alpha, L)$ conclui-se $\models_{\mathcal{LAPD}} \widehat{S^i(w)} \rightarrow [\alpha]\neg L$. Como $(w, w') \in \Upsilon_\alpha$, temos $\models_{\mathcal{LAPD}} \widehat{S^i(w)} \rightarrow \langle\alpha\rangle \widehat{S^i(w')}$, em particular $\models_{\mathcal{LAPD}} \widehat{S^i(w)} \rightarrow \langle\alpha\rangle L$, ou seja, $\models_{\mathcal{LAPD}} \widehat{S^i(w)} \rightarrow \neg[\alpha]\neg L$. Assim, juntando os resultados acima, teríamos $\models_{\mathcal{LAPD}} \widehat{S^i(w)} \rightarrow ([\alpha]\neg L \wedge \neg[\alpha]\neg L)$, donde segue que $\widehat{S^i(w)}$ é insatisfatível.

- Regra (RR): Suponhamos que S^i contém $w' :: L$, com $L \in LIT$, Υ^i_α contém (w, w') e que $w \models \neg Pre_{\mathcal{D}}(\alpha, L)$. Ao aplicar a regra (RR), deve-se adicionar $w :: L$ a S^i. Suponhamos que $\widehat{S^{i+1}(w)} = \widehat{S^i(w) \cup \{L\}}$ é insatisfatível. Então $\models_{\mathcal{LAPD}} \widehat{S^i(w)} \rightarrow \neg L$. Assim, de $w \models \neg Pre_{\mathcal{D}}(\alpha, L)$ conclui-se $\models_{\mathcal{LAPD}} \widehat{S^i(w)} \rightarrow [\alpha]\neg L$. Como $(w, w') \in \Upsilon_\alpha$, temos que $\models_{\mathcal{LAPD}} \widehat{S^i(w)} \rightarrow \langle\alpha\rangle \widehat{S^i(w')}$, em particular $\models_{\mathcal{LAPD}} \widehat{S^i(w)} \rightarrow \langle\alpha\rangle L$, ou seja, $\models_{\mathcal{LAPD}} \widehat{S^i(w)} \rightarrow \neg[\alpha]\neg L$. Com isso, temos $\models_{\mathcal{LAPD}} \widehat{S^i(w)} \rightarrow ([\alpha]\neg L \wedge \neg[\alpha]\neg L)$, donde segue que $\widehat{S^i(w)}$ é insatisfatível.

■

Assim, temos o seguinte teorema que estabelece a adequação de todo $\mathcal{LAPD}$-tableau:

Teorema 8.3.1 (Adequação) Se um $\mathcal{LAPD}$-tableau para A é fechado, então A é insatisfatível.

Prova: O lema 8.3.1 garante que cada uma das regras de tableau preserva a $\mathcal{LAPD}$-satisfatibilidade durante o processo de transformação de uma floresta em outra. Assim, se A é $\mathcal{LAPD}$-satisfatível, começa-se com uma floresta $\{\langle w_0 :: A, \emptyset\rangle\}$. Então, pelo lema 8.3.1, não é possível obter árvores $\mathcal{LAPD}$-insatisfatíveis com a aplicação das regras de tableau, não sendo possível, portanto, fechar o mesmo. ■

8.4 Completude

Nesta seção, provaremos a completude do método de tableau para $\mathcal{LAPD}$. A idéia fundamental consiste em mostrar como, a partir de um dado $\mathcal{LAPD}$-tableau aberto para uma fórmula A, pode-se construir um $\mathcal{LAPD}$-modelo para A. Por questões de clareza, abreviaremos $\{(w, w') : w \stackrel{\alpha}{\to} w'\}$ por Υ_α e $\bigcup_{\alpha \in A\c{C}\tilde{O}ES} \Upsilon_\alpha$ por $\Upsilon_{A\c{C}\tilde{O}ES}$.

Seja $\mathcal{T} = \mathcal{F}^\infty$ um tableau aberto (possivelmente infinito). Como $\mathcal{T}$ é aberto, existe uma árvore aberta $\langle S, \Upsilon\rangle$ em $\mathcal{T}$. Seja $\mu = \langle W, \{R_\alpha : \alpha \in A\c{C}\tilde{O}ES\}, R_\Box, \tau\rangle$ o modelo de Kripke definido como segue:

- W é o conjunto de mundos possíveis referenciados nas fórmulas rotuladas de S;
- $R_\alpha = \Upsilon_\alpha$ para $\alpha \in A\c{C}\tilde{O}ES$;
- $R_\Box = (\Upsilon_{A\c{C}\tilde{O}ES} \cup \Upsilon_\Box)^*$ (fecho reflexivo e transitivo de $\Upsilon_{A\c{C}\tilde{O}ES} \cup \Upsilon_\Box$);
- $w \in \tau(L)$ sse $w :: L \in S$.

Lema 8.4.1 μ é um $\mathcal{LAPD}$-modelo.

Prova:

- Pela construção de μ, temos que para toda ação $\alpha \in A\c{C}\tilde{O}ES$, $R_\alpha \subseteq R_\Box$. Também pela construção de μ, temos que $R_\Box$ é reflexiva e transitiva;
- Seja $(w, w') \in R_\alpha$. Devemos mostrar que

  1. se $w \models \neg Pre_{\mathcal{D}}(\alpha, L)$, então $w \notin \tau(L)$ implica $w' \notin \tau(L)$;
  2. se $w \models \neg Pre_{\mathcal{D}}(\alpha, \neg L)$, então $w \in \tau(L)$ implica $w' \in \tau(L)$.

  (Provando 1) Seja $w \notin \tau(L)$ e suponhamos que $w \models \neg Pre_{\mathcal{D}}(\alpha, L)$. Assim, temos $w :: L \notin S$ e $w \not\models C$ para todo C tal que α may cause L if C. Com isso, conclui-se $w' :: L \notin S$, pois do contrário teríamos $w :: L \in S$, pela regra (RR). Disso segue $w' \notin \tau(L)$.

(Provando 2) Seja $w \in \tau(L)$ e suponhamos que $w \models \neg Pre_{\mathcal{D}}(\alpha, \neg L)$. Assim, temos $w :: L \in S$ e $w \not\models C$ para todo C tal que α may cause $\neg L$ if C. Com isso, concluímos $w' :: L \in S$, pela regra (RP). Disso segue $w' \in \tau(L)$.

■

Resta agora demonstrar que o modelo μ acima satisfaz A.

Lema 8.4.2 Se B é uma subfórmula de A e $w :: B \in S$, então $\models^{\mu}_{w} B$.
Prova: Por indução na estrutura de B.
Passo base: se B é um literal, então o lema segue pela definição de τ.
Passo de indução:

- Os casos booleanos são como de costume;
- B é $[\alpha]B'$. Então $w :: [\alpha]B' \in S$. Assim, pela regra (K_{α}), $w' :: B' \in S$ para todo w' tal que $(w, w') \in R_{\alpha}$. Pela hipótese de indução, $\models^{\mu}_{w'} B'$, para todo w' tal que $(w, w') \in R_{\alpha}$. Disso segue $\models^{\mu}_{w} [\alpha]B'$.
- B é $\Box B'$. Primeiramente, $(w, w') \in R_{\Box}$ implica $(w, w') \in (\Upsilon_{AÇÕES} \cup \Upsilon_{\Box})^*$. Assim, $\exists w_0 = w, \ldots, w_k = w'$ tais que $(w_i, w_{i+1}) \in \Upsilon_{AÇÕES} \cup \Upsilon_{\Box}$ para $0 \leq i \leq k$.
Suponhamos agora $w :: \Box B' \in S$. Então $w_j :: \Box B' \in S$ para todo $0 \leq j \leq k$ por sucessivas aplicações da regra $(4_{\Box})$. Assim, $w' :: B' \in S$ pela regra $(T_{\Box})$. Pela hipótese de indução, temos que $\models^{\mu}_{w'} B'$. Com isso, conclui-se $\models^{\mu}_{w} B'$.
- B é $\neg\Box B'$. Então $w :: \neg\Box B' \in S$. Assim, pela regra $(\Diamond)$, existe w' tal que $(w, w') \in R_{\Box}$ e $w' :: \neg B' \in S$. Pela hipótese de indução, existe w' tal que $(w, w') \in R_{\Box}$ e $\models^{\mu}_{w'} \neg B'$, donde segue $\models^{\mu}_{w} \neg\Box B'$.
- B é $\neg[\alpha]B'$. A prova é análoga à do caso $B = \neg\Box B'$.

■

Com isso, temos o seguinte:

Lema 8.4.3 $\models^{\mu}_{w_0} A$
Prova: Dado um $\mathcal{LAPD}$-tableau para A, sua árvore inicial é $\langle w_0 :: A, \emptyset\rangle$, donde segue $w_0 :: A \in S$. Pelo lema 8.4.2, temos $\models^{\mu}_{w_0} A$. ■

Finalizando, temos o teorema que estabelece a completude do método de tableau aqui definido para $\mathcal{LAPD}$:

Teorema 8.4.1 (Completude) Se há um $\mathcal{LAPD}$-tableau aberto para A, então A é satisfatível.
Prova: Juntando os lemas 8.4.1, 8.4.2 e 8.4.3, segue o resultado. ■

Com isso, temos que o método de tableau para $\mathcal{LAPD}$ constitui-se num mecanismo de prova adequado e completo para a realização de inferências em modelagens de um mundo dinâmico.

8.5 Decidibilidade e terminação

De nada adianta um procedimento de inferência ser adequado e completo se ele estiver sujeito ao problema da parada. Nesta seção, apresentamos os aspectos que garantem a decidibilidade e a terminação do método de tableau discutido ao longo do presente capítulo.

Definição 8.5.1 A satisfatibilidade de uma dada fórmula A é $\mathcal{CLAPD}$-decidível se um $\mathcal{LAPD}$-tableau para $\neg A$ fecha no caso de A ser válida ou constrói um modelo finito no caso de A não ser. ■

Definição 8.5.2 Uma regra de tableau possui a *propriedade da subfórmula* sse toda fórmula em seu denominador for uma subfórmula de alguma fórmula no numerador. ■

Teorema 8.5.1 (Decidibilidade) A satisfatibilidade em $\mathcal{LAPD}$ é $\mathcal{CLAPD}$-decidível.
Prova: (Esboço) Todas as regras de tableau definidas na figura 8.1 possuem a propriedade da subfórmula. Sendo assim, somente um número finito de fórmulas distintas pode ser gerado durante o processo de prova, de maneira que ou o tableau em algum momento fecha, no caso de A ser satisfatível, ou construirá um modelo finito, no caso de A não o ser, donde segue o resultado. ■

Importante ressaltar que o teorema acima somente é relevante para aplicações práticas se se garantir que uma prova em uma dada implementação de $\mathcal{LAPD}$-tableau sempre termina. Devido ao fato de $\mathcal{LAPD}$ comportar o sistema de lógica monomodal $S4$, a realização de inferências nessa lógica está sujeita à ocorrência de *loops* ocasionados por sucessivas aplicações da regra $(4_\Box)$. Entretanto, mediante a implementação de um simples e eficiente teste de *loop* [31], pode-se garantir a terminação de qualquer implementação de $\mathcal{LAPD}$-tableau.

No próximo capítulo, promovemos uma análise comparativa da abordagem apresentada nesta parte com as principais propostas presentes na literatura, a fim de ilustrar as características mais importantes de $\mathcal{LAPD}$ e as vantagens de seu formalismo.

Parte IV

Discussões

Capítulo 9

Comparação com outras abordagens

Quem sou eu comparado ao universo?

— Ludwig van Beethoven

Neste capítulo, discutimos acerca das características da proposta apresentada nos capítulos 7 e 8 e a comparamos brevemente aos demais trabalhos analisados na parte II desta monografia.

9.1 Análise geral

Além de $\mathcal{LAPD}$ e $\mathcal{LAP}_{\leadsto}$, outras variantes de relação de dependência foram estudadas nos últimos anos. É o caso da dependência binária entre ações e átomos, apresentada em [6], e a dependência contextual, também entre ações e átomos, definida em [9].

As razões de se optar por uma ou outra de tais formas de dependência dizem respeito ao grau de intuitividade da fase de geração do domínio em cada um de tais formalismos. Observando os resultados obtidos nos trabalhos supracitados, consideramos que uma dependência envolvendo literais é mais intuitiva.

No tocante à representação de efeitos indiretos, conforme observado em [8] e verificado nos capítulos 3–6, os trabalhos [37, 38, 64, 65, 7] apóiam a idéia de que o conceito de causalidade explícita é realmente uma noção fundamental para a solução do problema da ramificação (lembrando que a abordagem de fecho de explicação, analisada no capítulo 5, utiliza uma noção causal implícita, a qual traz dificuldades para a representação de domínios envolvendo efeitos indiretos em termos de modularidade — cf. critério 3 da seção 2.5). Entretanto, uma noção causal contextualizada se faz necessária a fim de podermos obter uma melhor solução para certos cenários [4]. Isso serve para apoiar ainda mais o trabalho aqui desenvolvido.

Quanto à simplicidade e clareza do formalismo de base para representação de domínios, consideramos $\mathcal{LAPD}$ uma lógica bastante clara e simples, com uma semântica bem definida e flexível e com um grande poder expressivo. Isso lhe confere um alto grau de vantagem em relação à maioria das abordagens propostas na literatura ao longo dos últimos anos.

Na próxima seção, discutimos acerca das características monotônicas de nossa solução.

9.2 Monotonicidade

Na realidade, a distinção entre abordagens monotônicas e não-monotônicas não é precisa, necessitando de uma melhor definição [13]. Com efeito, Shanahan [63] mostra que mesmo algumas propostas consideradas monotônicas, como [56], utilizam por base uma espécie de raciocínio não-monotônico no seguinte aspecto. Seja $Exp(LEIS_0)$ o fecho de explicação[1] de uma dada teoria de ações $LEIS_0$. Ao invés de considerar a derivação monotônica $Exp(LEIS_0) \vdash A$, Shanahan examina a relação de inferência $LEIS_0 \vdash A$. Assim, pode-se mostrar que $\vdash$ é não-monotônica, pois é possível que se tenha $Exp(LEIS_0) \vdash A$ sem necessariamente se ter $Exp(LEIS_0 \wedge LEIS_1) \vdash A$, para algum conjunto de fórmulas $LEIS_1$.

Seguindo o mesmo raciocínio, nossa solução pode ser considerada monotônica. A justificativa para tal é fundamentada no seguinte fato: $LEIS_0 \vdash A$ no caso de $\mathcal{LAPD}$ é definida como $\models_{\mathcal{LAPD}} LEIS_0 \rightarrow A$, assim, $LEIS_0 \vdash A$ implica $LEIS_0 \wedge LEIS_1 \vdash A$. O ponto fundamental aqui é o fato de a nossa noção causal estar no nível de uma metalinguagem, ao contrário das propostas presentes em [61, 62], que a expressam na própria linguagem-objeto.

Entretanto, é importante ressaltar que, se por um lado nossa lógica de base é monotônica, por outro alterações na relação de dependência $\mathcal{D}$ podem mudar o conjunto de conclusões deriváveis de uma teoria, fazendo com que nossa abordagem possa também ser considerada não-monotônica com respeito a relações de dependência.

9.3 Problema da persistência

Observando cada uma das abordagens analisadas nos capítulos 3–6, pode-se perceber que todas elas resolvem de maneira razoavelmente satisfatória o problema da persistência. Abstraindo-se o problema da ramificação, a questão relacionada a domínios mais complexos e a eventual necessidade de se enunciar axiomas de persistência condicionais, pode-se considerar que o problema da persistência já está praticamente resolvido.

O ponto fraco da grande maioria das abordagens estudadas, conforme argumentado ao

[1] Ou seja, $LEIS_0$ acrescida do conjunto de axiomas de fecho de explicação necessários, nos moldes do capítulo 5.

longo deste trabalho, reside no fato de serem fundamentadas em um formalismo de base indecidível e de não se comportarem adequadamente em domínios mais elaborados. Nesse aspecto, $\mathcal{LAPD}$ apresenta as vantagens de ser decidível, comportar-se bem em domínios complexos e possuir um método de inferência adequado e completo.

9.4 Leis de executabilidade

Assim como em $\mathcal{LAP}_{\leadsto}$, no formalismo de $\mathcal{LAPD}$ somos obrigados a enunciar leis de executabilidade, como, por exemplo, $\Box\langle esperar\rangle\top$, $\Box\langle carregar\rangle\top$ e $\Box\langle atirar\rangle\top$ no exemplo 7.6.1. É importante ressaltar, porém, que as abordagens estudadas também se vêem diante dessa dificuldade, como é o caso do formalismo de Lin, que utiliza o predicado *Poss* (cf. seções 2.4 e 3.3).

Uma maneira de livrar o projetista do sistema dessa tarefa é gerar tais leis de executabilidade de forma automática. Neste caso, utiliza-se a hipótese de que todas as ações são maximalmente executáveis. É exatamente através dessa hipótese que em [4] é proposto um algoritmo para geração automática das leis de executabilidade a partir das leis estáticas em conjunção com as leis de ações. Sendo assim, uma modificação do procedimento definido naquele trabalho pode ser aplicado em $\mathcal{LAPD}$ para a obtenção das leis de executabilidade.

9.5 Efeitos indiretos

A relação de dependência ternária $\mathcal{D}$ é uma forma de conexão causal fraca entre ações e literais, restrita por um determinado contexto. Sendo assim, a abordagem proposta neste trabalho segue a linha das soluções mais recentes ao problema da ramificação, no sentido de que provê uma noção causal para o processo de descrição do domínio.

Entretanto, conforme pode ser observado nos cenários-exemplos da seção 7.6, se por um lado não precisamos enunciar os efeitos indiretos das ações, por outro somos obrigados a escrever dependências indiretas, de maneira que nossa abordagem está sujeita às mesmas críticas feitas ao trabalho apresentado no capítulo 6. No entanto, um argumento a favor desse procedimento é o fato de que as propostas presentes na literatura que não efetuam o enunciado de efeitos indiretos acabam sendo restritas a apenas algumas classes de problemas. Como exemplo, a abordagem apresentada em [56] limita-se a domínios com ações determinísticas e sem efeitos indiretos, enquanto que a presente em [13] não é capaz de tratar ações com efeitos ao mesmo tempo indiretos e não-determinísticos [9].

Com o enunciado de dependências indiretas, é possível tratar exemplos problemáticos para outras abordagens, como é o caso do cenário da porta bloqueada, o qual não é corretamente resolvido nas abordagens de minimização da causalidade e de pós-derivação de efeitos sem que se enunciem explicitamente efeitos indiretos na lógica de base.

9.6 Ações não-determinísticas

Na verdade, não existe uma linha divisória bem definida separando ações determinísticas de ações não-determinísticas. As diferenças são bastante sutis e dependem muito do nível de descrição considerado, de maneira que os efeitos de uma ação podem ser determinados em um dado nível de descrição mas indeterminados em outro [38].

Um formalismo de base que considere ações determinísticas e não-determinísticas de maneiras fundamentalmente diferentes pode esbarrar em dificuldades relacionadas a eventuais mudanças da linguagem. Nesse aspecto, $\mathcal{LAPD}$ constitui um bom formalismo de base no sentido em que pode ser usada no seu todo para a representação de ambos os tipos de domínio, sem promover distinções no nível sintático para cada situação.

Lin [38] propõe uma maneira de se tratar domínios envolvendo ações não-determinísticas através de uma extensão do método analisado no capítulo 3. Entretanto, essa abordagem está sujeita aos mesmos inconvenientes discutidos no referido capítulo, notadamente por dar margem a situações em que se faz necessário o uso de lógica de segunda ordem.

Para avaliar o comportamento de $\mathcal{LAPD}$ em domínios com ações não-determinísticas, apresentamos a representação em nossa abordagem do cenário do tiro em Toulouse [20]. Tal cenário ilustra uma instância de domínios com não-determinismo, pois, estando a arma carregada mas com o tambor não completo, após a execução de *atirar*, pode-se ter como resultado que a vítima estará morta bem como que ela continuará viva. Observe que em $\mathcal{LAPD}$ a representação desse cenário e as conclusões que dele podem ser obtidas estão de acordo com o esperado:

Exemplo 9.6.1 (Seqüência do exemplo 6.5.1) A modelagem do cenário do tiro em Toulouse é formalizada em $\mathcal{LAPD}$ da seguinte maneira:

$$LEIS = \left\{ \begin{array}{l} \Box\langle esperar\rangle\top, \\ \Box\langle carregar\rangle\top, \\ \Box\langle atirar\rangle\top, \\ \Box(Cheio \rightarrow Carregada), \\ \Box[carregar]Carregada, \\ \Box[atirar]\neg Cheio, \\ \Box(Cheio \rightarrow [atirar]\neg Vivo) \end{array} \right\}$$

$$\mathcal{D} = \left\{ \begin{array}{c} atirar \ \textsf{may cause} \ \neg Vivo \ \textsf{if} \ Carregada, \\ atirar \ \textsf{may cause} \ \neg Cheio \ \textsf{if} \ \top, \\ carregar \ \textsf{may cause} \ Cheio \ \textsf{if} \ \neg Cheio \end{array} \right\}$$

A figura 9.1 a seguir ilustra o não-determinismo presente neste cenário, o qual é capturado pela representação em $\mathcal{LAPD}$ acima. ■

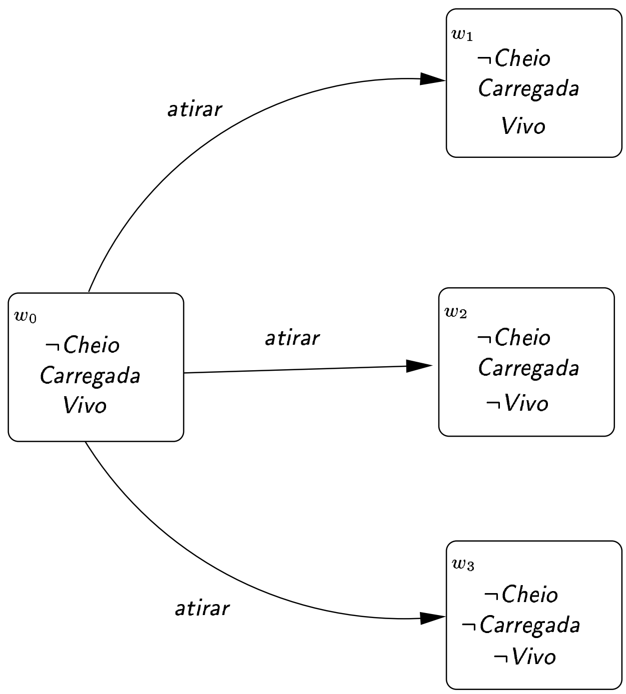

Figura 9.1: Não determinismo da ação *atirar* no cenário do tiro em Toulouse.

Considerando domínios envolvendo ações não-determinísticas em conjunção com efeitos indiretos, temos que $\mathcal{LAPD}$ também se mostra adequada para representação e realização de inferências, conforme ilustrado pelo exemplo a seguir.

Exemplo 9.6.2 (O cenário das Mailboxes [9]) Suponha que $Mbox_1$ significa "o e-mail está na mailbox 1" e que $Mbox_2$ quer dizer "o e-mail está na mailbox 2", e considere as ações $salvar_1$ e $salvar_2$, que sempre salvam uma mensagem de e-mail em $Mbox_1$ e em $Mbox_2$, respectivamente. Suponha também que há uma ação não-determinística *salvar* que salva o e-mail em uma das duas mailboxes ou em ambas. Representamos o fato de que o e-mail está salvo em $Mbox_1$ ou em $Mbox_2$ ou em ambas pelo literal *Salvo*. A representação deste cenário em $\mathcal{LAPD}$ é a seguinte ($i = 1, 2$):

$$LEIS = \left\{ \begin{array}{l} \Box\langle salvar\rangle\top, \Box\langle salvar_i\rangle\top, \\ \Box(Salvo \leftrightarrow Mbox_1 \vee Mbox_2), \\ \Box[salvar]Salvo, \\ \Box[salvar_i]Mbox_i \end{array} \right\}$$

$$\mathcal{D} = \left\{ \begin{array}{c} salvar \text{ may cause } Salvo \text{ if } \top, \\ salvar \text{ may cause } Mbox_i \text{ if } \top, \\ salvar_i \text{ may cause } Mbox_i \text{ if } \top, \\ salvar_i \text{ may cause } Salvo \text{ if } \top \end{array} \right\}$$

$$BC = \{\neg Salvo, \neg Mbox_1, \neg Mbox_2\}$$

A partir da representação acima, pode-se concluir

$$\models_{\mathcal{LAPD}}(BC \wedge LEIS) \rightarrow [salvar](Mbox_1 \vee Mbox_2)$$

que está de acordo com o esperado. ■

Esse último exemplo, na verdade, é problemático para todas as abordagens existentes (i.e. abordagens para tratar ações não-determinísticas com efeitos indiretos). Acreditamos que o problema seja o seguinte: se temos efeitos indiretos não-determinísticos, então

1. necessariamente temos de isentar tais efeitos do processo de minimização (no sentido de evitar interpretação exclusiva de disjunções inclusivas, cf. o clássico exemplo de lançar uma moeda no tabuleiro de xadrez[2] de Reiter);

2. como o efeito é indireto, essa isenção deve ser especificada indiretamente: mencionando o contexto no qual ela se aplica, sem mencionar, porém, a ação.

Sendo assim, é difícil garantir que nenhuma outra ação se aplica exatamente na mesma circunstância, sem o enunciado indireto explícito.

Além desses detalhes, a utilização da abordagem de Lin [38] para representação desse cenário nos leva a concluir que a execução da ação $salvar_1$ provoca uma alteração em $Mbox_2$, o que não é intuitivo.

[2]Do original em inglês *dropping a coin on a chessboard.*

Capítulo 10

Conclusões

Por tudo aquilo que foi e o que poderia ter sido,
bendita a graça de ser.

— Helena Kolody

No decorrer do presente trabalho, tivemos a oportunidade de perceber a importância da área de raciocínio sobre ações para que os objetivos da Inteligência Artificial possam ser atingidos. Elaboramos as motivações e definimos operacionalmente os problemas da persistência, da ramificação e da qualificação, cuja solução é de vital importância para os propósitos da área.

A fim de estabelecer o atual panorama das pesquisas em torno desses problemas, elaboramos também uma análise das principais e mais completas soluções apresentadas até então, promovendo uma discussão crítica a respeito dos ganhos trazidos à comunidade assim como dos aspectos que ainda precisam ser melhorados.

Sendo assim, por ainda haver lacunas presentes nos formalismos propostos na literatura, propusemos também a nossa abordagem, fundamentada no princípio de causalidade explícita, seguindo um paradigma monotônico para raciocinar sobre ações. Argumentamos que nosso formalismo, além de se adequar perfeitamente às tarefas para as quais as demais soluções foram propostas, acrescenta ainda sua parcela de contribuição para as pesquisas desenvolvidas na área. Isso se dá no sentido de que as idéias aqui apresentadas aproveitam muitos dos pontos positivos presentes em outras abordagens, acrescentando-lhes melhorias, e ainda oferecem possibilidades de futuras extensões.

Finalizamos este trabalho apresentando, na seqüência, as principais conclusões que dele podem ser obtidas, considerando o que foi exposto ao longo dos capítulos 1–9, e discutimos acerca das possíveis extensões que podem nortear futuros trabalhos na área.

Utilizando o formalismo de $\mathcal{LAPD}$, podemos realizar tarefas de predição, explicação e planificação em um mundo dinâmico, tais quais definidas na seção 1.2, o que satisfaz o critério 4, estabelecido na seção 2.5.

Com a definição da dependência ternária, que torna possível capturar os contextos nos quais uma determinada ação é executada, pode-se obter uma representação ainda mais econômica do domínio em consideração, o que está de acordo com o critério 1.

Conforme apresentado nas seções 7.6 e 9.6, $\mathcal{LAPD}$ pode ainda ser utilizada para a modelagem de cenários envolvendo efeitos indiretos e ações não-determinísticas. Nesse caso, basta que se enunciem as devidas dependências indiretas. Outras abordagens presentes na literatura, como, por exemplo, [38] e [24], também necessitam que se enunciem explicitamente dependências indiretas para tratar corretamente esses domínios. Sendo assim, concluímos que nossa abordagem está de acordo com o critério 2, estabelecido na seção 2.5.

Com relação ao critério 3, apesar de o mesmo, conforme comentado na seção 2.5, ser discutível, concluímos que $\mathcal{LAPD}$ apresenta um grau de modularidade bastante razoável. A justificativa para tal reside no fato de que toda e qualquer eventual mudança a ser empreendida na representação do domínio será concentrada ou nas fórmulas da lógica de base ou nos elementos da relação de dependência, de maneira completamente independente do mecanismo de inferência.

Ao contrário da grande maioria das abordagens presentes na literatura, procuramos focar nosso objetivo também na definição de um mecanismo de inferência capaz de prover um caráter prático à solução apresentada. Nesse sentido, definimos um método de prova para $\mathcal{LAPD}$ baseado nos sistemas de tableau semânticos e provamos que o mesmo é adequado, completo e decidível, satisfazendo o critério 5 da seção 2.5. Em termos de implementação, nosso método pode ser facilmente implementado através do sistema definido em [57], o qual consiste de uma extensão do trabalho desenvolvido em [67].

No tocante às principais contribuições trazidas por este trabalho, podemos resumi-las nos seguintes itens:

- Empreendemos uma comparação sistemática das principais abordagens causais propostas até então, compilando suas idéias de base, ilustrando suas diferenças e apontando suas lacunas. Em especial, fizemos uma análise mais detalhada da abordagem de minimização da causalidade, focando seu mecanismo de inferência.

- Propomos raciocinar sobre ações usando como base o elegante formalismo da lógica de ações e planos $\mathcal{LAP}$ [4], acrescentando-lhe uma relação de dependência ternária como forma de expressar a noção de causalidade.

- Em nosso formalismo, não há a necessidade de se expressar explicitamente informações sobre os não-efeitos das ações, ponto fundamental que difere nosso trabalho do de [4].

- Mostramos que a relação de dependência ternária $\mathcal{D}$ oferece um maior grau de concisão em comparação à dependência binária $\leadsto$ sem promover aumento de complexidade algum.

- Apresentamos a modelagem de vários cenários-exemplos, cujos resultados apóiam a hipótese de que não há como se obter uma solução definitiva para o problema da ramificação sem se enunciar algumas dependências indiretas.

- Apresentamos um método de inferência adequado, completo e decidível, baseado no método de tableau proposto em [5].

Uma eventual crítica que poderia ser feita ao nosso método diz respeito à complexidade computacional apresentada por implementações para realização de inferências em $\mathcal{LAPD}$. Entretanto, é importante ressaltar que essa é uma dificuldade inerente ao próprio problema da satisfatibilidade em $\mathcal{LAPD}$, que é EXPTIME, e não de nosso método, de maneira que qualquer outra solução proposta também a ela estará sujeita. A vantagem de nosso método de prova, entretanto, é que ele é realmente viável.

Uma outra crítica que também poderia ser feita ao trabalho aqui apresentado é a necessidade de, tal qual em $\mathcal{LAP}_{\leadsto}$, ter de se enunciar leis de executabilidade na representação do domínio. No entanto, com a utilização dos algoritmos que estão sendo implementados em [49], pode-se automatizar essa fase da descrição do domínio.

Com relação a trabalhos futuros, seria interessante a modificação do mecanismo supervisionado para a geração da relação de dependência proposto em [4] e que está sendo implementado em [49] no sentido de gerar a dependência ternária de forma interativa.

Uma área em que a utilização de $\mathcal{LAPD}$ pode ser interessante é a modelagem de diálogos entre seres humanos e computadores, conforme empreendido nos trabalhos subseqüentes de [15]. Nesse sentido, possíveis extensões de $\mathcal{LAPD}$ deveriam redefini-la de maneira a comportar também operadores para representar os conceitos de tempo e crença, o que ampliaria a gama de aplicações de nosso formalismo a um conjunto maior de situações, principalmente as de interesse prático.

Referências Bibliográficas

[1] Carlos Alchourrón, Peter Gärdenfors, e David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50:510–530, 1985.

[2] Andrew B. Baker. Nonmonotonic reasoning in the framework of the situation calculus. *Artificial Intelligence*, 49(1–3):5–23, 1991.

[3] Patrick Blackburn e Johan Bos. Representation and inference for natural language: A first course in computational semantics. `http://www.comsem.org`, 1999.

[4] Marcos A. Castilho. *Modèles logiques pour le raisonnement sur les actions.* Tese de Doutorado, Université Paul Sabatier, Toulouse, outubro de 1998.

[5] Marcos A. Castilho, Luis Fariñas del Cerro, Olivier Gasquet, e Andreas Herzig. Modal tableaux with propagation rules and structural rules. *Fundamenta Informaticae*, 32(3–4):281–297, 1997.

[6] Marcos A. Castilho, Olivier Gasquet, e Andreas Herzig. Modal tableaux for reasoning about actions and plans. S. Steel e R. Alami, editores, *Proc. of ECP'97*, número 1348 in LNAI, pág. 104–116, Toulouse, setembro de 1997.

[7] Marcos A. Castilho, Olivier Gasquet, e Andreas Herzig. Formalizing action and change in modal logic I: the frame problem. *Journal of Logic and Computation*, 9(5):701–735, 1999.

[8] Marcos A. Castilho, Andreas Herzig, e Camilla Schwind. Raisonnement sur les actions: les approches basées sur la causalité et la dépendance. R. Jeansoulin, editor, *Nouveaux défis en sciences de l'information: documents et évolution*, Toulouse, setembro de 2000.

[9] Marcos A. Castilho, Andreas Herzig, e Ivan J. Varzinczak. It depends on the context! a decidable logic of actions and plans based on a ternary dependence relation. S. Benferhat e E. Giunchiglia, editores, *Proc. of NMR'2002*, pág. 343–348, Toulouse, 2002.

[10] Brian F. Chellas. *Modal Logic: an introduction.* Cambridge University Press, 1980.

[11] Keith Clark. Negation as failure. H. Gallaire e J. Minker, editores, *Logics and Databases*, pág. 293–322. New York, 1978.

[12] William F. Clocksin e Christopher S. Mellish. *Programming in Prolog.* Springer-Verlag, Berlin, terceira edition, 1987.

[13] Patrick Doherty, Witold Łukaszewicz, e Andrzej Szałas. Explaining explanation closure. *Proc. Int. Symp. on Methodologies for Intelligent Systems*, Zakopane, 1996.

[14] Luis Fariñas del Cerro, David Fauthoux, Olivier Gasquet, Andreas Herzig, Dominique Longin, e Fabio Massacci. Lotrec: the generic tableau prover for modal and description logics. *Proc. of IJCAR'2001*, LNCS, pág. 6, junho de 2001.

[15] Luis Fariñas del Cerro, Andreas Herzig, Dominique Longin, e Omar Rifi. Belief reconstruction in cooperative dialogues. *Artificial Intelligence: Methodology, Systems, Applications*, pág. 254–266, 1998.

[16] Richard E. Fikes e Nils J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[17] Jeff J. Finger. *Exploiting Constraints in Design Synthesis.* Tese de Doutorado, Department of Computer Science — Stanford University, Stanford, CA, 1987.

[18] Melvin Fitting. *Proof Methods for Modal and Intuitionistic Logics.* D. Reidel, Dordrecht, 1983.

[19] Peter Gärdenfors. *Knowledge in Flux: Modeling the Dynamics of Epistemic States.* The MIT Press, 1988.

[20] Olivier Gasquet e Andreas Herzig. Reasoning about actions using dependence relations. Relatório técnico, IRIT — Université Paul Sabatier, Toulouse, 1995.

[21] Michael Gelfond e Vladimir Lifschitz. Representing action and change by logic programs. *Journal of Logic Programming*, 17:301–322, 1993.

[22] Matthew L. Ginsberg. A circumscriptive theorem prover. *Artificial Intelligence*, 39:209–230, 1989.

[23] Matthew L. Ginsberg e David E. Smith. Reasoning about actions II: The qualification problem. *Artificial Intelligence*, 35(3):311–342, 1988.

[24] Enrico Giunchiglia, G. Neelakantan Kartha, e Vladimir Lifschitz. Representing action: indeterminacy and ramifications. *Artificial Intelligence*, 95:409–438, 1997.

[25] Rajeev Goré. Tableaux methods for modal and temporal logics. M. D'Agostino et al., editor, *Handbook of Tableaux Methods*, pág. 297–396. Kluwer, 1999.

[26] Andrew R. Haas. The case for domain-specific frame axioms. F. M. Brown, editor, *Proc. of the 1987 Workshop on the Frame Problem*, pág. 343–348, 1987.

[27] Steve Hanks e Drew McDermott. Default reasoning, nonmonotonic logics and the frame problem. *Proc. of AAAI'86*, pág. 328–333, Philadelphia, NJ, 1986.

[28] David Harel. Dynamic logic. D. M. Gabbay e F. Guenther, editores, *Handbook of Philosophical Logic*, volume 2: Extensions of Classical Logic, pág. 497–604. D. Reidel Publishing Co., Dordrecht, 1984.

[29] David Harel. *Algorithmics: The Spirit of Computing.* Addison Wesley, segunda edition, 1993.

[30] Leon Henkin. The completeness of the first-order functional calculus. *Journal of Symbolic Logic*, 14:159–166, 1949.

[31] Alain Heuerding, Michael Seyfried, e Heinrich Zimmermann. Efficient loop-check for backward proof search in some non-classical propositional logics. *Proc. of Tableaux'96*, número 1071 in LNAI, pág. 210–225, 1996.

[32] George E. Hughes e Max J. Cresswell. *An introduction to modal logic.* Methuen and Co. Ltd., London, 1968.

[33] George E. Hughes e Max J. Cresswell. *A companion to modal logic.* Methuen and Co. Ltd., London, 1984.

[34] Saul A. Kripke. A completeness theorem in modal logic. *Journal of Symbolic Logic*, 24(1):1–14, 1959.

[35] Vladimir Lifschitz. Frames in the space of situations. *Artificial Intelligence*, 46:365–376, 1990.

[36] Vladimir Lifschitz. Circumscription. D. M. Gabbay, C. J. Hogger, e J. A. Robinson, editores, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 3, pág. 297–352. Oxford University Press, 1994.

[37] Fangzhen Lin. Embracing causality in specifying the indirect effects of actions. C. Mellish, editor, *Proc. of IJCAI'95*, pág. 1985–1991, Montreal, 1995.

[38] Fangzhen Lin. Embracing causality in specifying the indeterminate effects of actions. *Proc. of AAAI'96*, volume 1, pág. 670–676, 1996.

[39] Fangzhen Lin. On strongest necessary and weakest sufficient conditions. *Artificial Intelligence*, 128:143–159, 2001.

[40] Fangzhen Lin e Raymond Reiter. State constraints revisited. *Journal of Logic and Computation*, 4(5):655–677, 1994.

[41] Norman McCain e Hudson Turner. A causal theory of ramifications and qualifications. C. Mellish, editor, *Proc. of IJCAI'95*, pág. 1978–1984, Montreal, 1995.

[42] Norman McCain e Hudson Turner. Causal theories of action and change. *Proc. of AAAI'97*, pág. 460–465, 1997.

[43] John McCarthy. Programs with common sense. *Proc. of the Teddington Conf. on the Mechanization of Thought Processes*, pág. 75–91, London, 1959.

[44] John McCarthy. Epistemological problems of artificial intelligence. *Proc. of IJCAI'77*, pág. 1038–1044, 1977.

[45] John McCarthy. Circumscription — a form of non-monotonic reasoning. *Artificial Intelligence*, 13:27–39, 1980.

[46] John McCarthy. Applications of circumscription to formalizing common-sense knowledge. *Artificial Intelligence*, 28:89–116, 1986.

[47] John McCarthy e Patrick Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, 4:463–502, 1969.

[48] Elliott Mendelson. *Introduction to Mathematical Logic*. D. Van Nostrand Company, segunda edition, 1979.

[49] Razer N. R. Montaño. Geração semi-automática de descrições de domínio. Disponível em `http://www.inf.ufpr.br/~marcos/projetos`.

[50] Robert C. Moore. Semantical considerations on nonmonotonic logic. *Artificial Intelligence*, 25:75–94, 1985.

[51] Nils J. Nilsson. Shakey the robot. Relatório técnico, SRI, Menlo Park, CA, 1984. SRI Technical Note no. 323.

[52] Christos H. Papadimitriou. *Computational Complexity*. Addison Wesley, 1994.

[53] Sally Popkorn. *First Steps in Modal Logic*. Cambridge University Press, 1994.

[54] Teodor C. Przymusinski. An algorithm to compute circumscription. *Artificial Intelligence*, 38(1):49–73, 1989.

[55] Raymond Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[56] Raymond Reiter. The frame problem in the situation calculus: a simple solution (sometimes) and a completeness result for goal regression. V. Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation*, Papers in Honor of John McCarthy, pág. 359–380. Academic Press, 1991.

[57] Roberta V. Rojo. Uma implementação genérica para métodos de tableau modais com uma aplicação específica. Disponível em `http://www.inf.ufpr.br/~marcos/projetos`.

[58] Erik Sandewall. An approach to the frame problem, and its implementation. *Machine Intelligence*, 7:195–204, 1972.

[59] Erik Sandewall. *Features and Fluents: the Representation of Knowledge about Dynamical Systems*, volume 1. Oxford University Press, 1994.

[60] Erik Sandewall. Assessments of ramification methods that use static domain constraints. L. C. Aiello, J. Doyle, e S. Shapiro, editores, *Proc. of KR'96*, pág. 99–110, San Francisco, CA, 1996.

[61] Lenhart K. Schubert. Monotonic solution of the frame problem in the situation calculus: an efficient method for worlds with fully specified actions. H. E. Kyberg, R. P. Loui, e G. N. Carlson, editores, *Knowledge Representation and Defeasible Reasoning*, pág. 23–67, 1990.

[62] Lenhart K. Schubert. Explanation closure, action closure, and the Sandewall test suite for reasoning about change. *Journal of Logic and Computation*, 4(5):679–700, 1994.

[63] Murray Shanahan. *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia.* The MIT Press, Cambridge, MA, 1997.

[64] Michael Thielscher. Computing ramifications by postprocessing. C. Mellish, editor, *Proc. of IJCAI'95*, pág. 1994–2000, Montreal, 1995.

[65] Michael Thielscher. Ramification and causality. *Artificial Intelligence*, 89(1–2):317–364, 1997.

[66] Ivan J. Varzinczak. Algoritmos de circunscrição. Monografia de conclusão de curso de graduação disponível em `http://www.inf.ufpr.br/~ivan`, dezembro de 1999.

[67] Ivan J. Varzinczak. Provadores automáticos de teoremas para as lógicas monomodais proposicionais *K* e *S4*. Monografia de conclusão de curso de graduação disponível em `http://www.inf.ufpr.br/~ivan`, agosto de 2000.

Apêndice A

Adequação e completude de $\mathcal{LAPD}$

Apresentamos aqui a prova de que a axiomática de $\mathcal{LAPD}$ é adequada e completa com relação à semântica definida na seção 7.2.

A adequação de $\mathcal{LAPD}$ é imediata: dada a adequação de $\mathcal{LAP}$ [4], a única coisa a ser feita é constatar que todas as ocorrências de $Persist([\alpha])$ são válidas. A completude, por sua vez, é conseqüência direta do teorema A.1 abaixo.

A prova do teorema A.1, enunciado logo a seguir, segue o método tradicional de Henkin [30]. Nesse caso, precisaremos antes dos seguintes lemas:

Lema A.1 (Lema de Lindenbaum) Sendo a linguagem de $\mathcal{LAPD}$ enumerável, todo conjunto $\mathcal{LAPD}$-consistente S admite uma extensão $\mathcal{LAPD}$-maximal-consistente. ■

Lema A.2 Seja S um conjunto $\mathcal{LAPD}$-maximal-consistente e sejam A e B fórmulas.

1. Se $\vdash_{\mathcal{LAP}} A$, então $A \in S$;

2. $A \in S$ se e somente se $\neg A \notin S$;

3. $A \wedge B \in S$ se e somente se $A \in S$ e $B \in S$.

Os demais conectivos lógicos são definidos da maneira usual. ■

Ambos os lemas acima são resultados da lógica clássica (não-modal) proposicional. Para o caso modal, precisaremos ainda da seguinte definição e do lema logo a seguir:

Definição A.1 Seja S um conjunto de fórmulas de $\mathcal{LAP}$. Definimos $\Box^-(S)$ como sendo o conjunto $\{A : \Box A \in S\}$ e $[\alpha]^-(S)$ como sendo o conjunto $\{A : [\alpha]A \in S\}$. ■

Lema A.3 Seja S um conjunto de fórmulas $\mathcal{LAPD}$-consistente contendo uma fórmula $\Diamond A$ (respectivamente $\langle\alpha\rangle A$). Então o conjunto $\Box^-(S) \cup \{A\}$ (respectivamente $[\alpha]^-(S) \cup \{A\}$) é $\mathcal{LAPD}$-consistente.

Prova: Daremos a prova apenas para o caso $\Diamond A$ (o caso $\langle\alpha\rangle A$ é análogo). Suponhamos que $\Box^{-}(S) \cup \{A\}$ seja $\mathcal{LAPD}$-inconsistente. Neste caso, existe um subconjunto finito $\{\Box A_1, \ldots, \Box A_n\}$ de S tal que $\vdash_{\mathcal{LAP}} (A_1 \wedge \ldots \wedge A_n \wedge A) \rightarrow \bot$. Assim, $\vdash_{\mathcal{LAP}} (A_1 \wedge \ldots \wedge A_n) \rightarrow \neg A$, o que, pela regra de necessitação $RN(\Box)$, o axioma $K(\Box)$ e o *modus ponens*, resulta em $\vdash_{\mathcal{LAP}} \Box(A_1 \wedge \ldots \wedge A_n) \rightarrow \Box\neg A$, donde segue $\vdash_{\mathcal{LAP}} \Box(A_1 \wedge \ldots \wedge A_n) \rightarrow \neg\Diamond A$. Pela distributividade do operador $\Box$, temos $\vdash_{\mathcal{LAP}} (\Box A_1 \wedge \ldots \wedge \Box A_n) \rightarrow \neg\Diamond A$. Como cada $\Box A_i$ está em S, então, pelo lema A.2, $\neg\Diamond A$ deve também estar em S, o que leva à conclusão de que S é $\mathcal{LAPD}$-inconsistente, contra a hipótese. ∎

Em adição a esses resultados, sabendo-se que um modelo canônico para $\mathcal{LAPD}$ é $\mu = \langle W, \{R_\alpha : \alpha \in A\c{C}\tilde{O}ES\}, R_\Box, \tau\rangle$, onde:

- W é o conjunto de todos os conjuntos $\mathcal{LAPD}$-maximal-consistentes;
- $wR_\Box w'$ se e somente se $\Box^{-}(w) \subseteq w'$;
- para toda ação atômica α, $wR_\alpha w'$ se e somente se $[\alpha]^{-}(w) \subseteq w'$;
- para todo átomo P e todo mundo possível $w \in W$, $w \in \tau(P)$ se e somente se $P \in w$,

podemos enunciar o seguinte lema:

Lema A.4 Para toda fórmula A, $\models^{\mu}_{w} A$ se e somente se $A \in w$.
Prova: A prova é por indução na estrutura de A. Para átomos, o teorema é imediato, de acordo com a definição de τ. No passo de indução, os casos booleanos são triviais. Sendo assim, provaremos apenas o caso $A = [\alpha]B$ (o caso $A = \Box B$ é completamente análogo).

($\Rightarrow$): Suponhamos que $\models^{\mu}_{w} [\alpha]B$ e que $[\alpha]B \notin w$. Assim, pelo lema A.2, $\neg[\alpha]B \in w$ (i.e. $\langle\alpha\rangle\neg B \in w$). Seja $S = [\alpha]^{-}(S) \cup \{\neg B\}$. Então, pelo lema A.3, S é $\mathcal{LAPD}$-consistente e, pelo lema A.1, pode ser estendido para um conjunto $\mathcal{LAPD}$-maximal-consistente w'. Pela construção de μ, $w' \in W$ e $wR_\alpha w'$. Como $\neg B \in w'$, pela hipótese de indução, $\models^{\mu}_{w'} \neg B$ e portanto $\not\models^{\mu}_{w} [\alpha]B$, contra a hipótese.

($\Leftarrow$): Suponhamos que $[\alpha]B \in w$. Então, pela definição de R_α, temos que $B \in w'$ para todo w' tal que $wR_\alpha w'$. Pela hipótese de indução, temos $\models^{\mu}_{w'} B$, donde, pela definição de satisfação, segue $\models^{\mu}_{w} [\alpha]B$. ∎

Precisaremos ainda provar que μ é um $\mathcal{LAPD}$-modelo:

Lema A.5 O modelo canônico definido acima é um $\mathcal{LAPD}$-modelo.
Prova: A prova de que $R_\Box$ é reflexiva e transitiva é feita da maneira usual. Sendo assim, basta provarmos:

1. $R_\alpha \subseteq R_\Box$ para todo $\alpha \in A\c{C}\tilde{O}ES$;
2. Se, para todo $C \in FORPROP$ tal que α may cause L if C, $w \not\models C$, então $L \in w'$ implica $L \in w$.

(Provando 1.): Temos que provar que se $wR_\alpha w'$, então $wR_\Box w'$, ou seja $\Box^-(w) \subseteq w'$. Seja $A \in \Box^-(w)$. Isso vale se e somente se $\Box A \in w$, e, como $\vdash_{\mathcal{LAP}} \Box A \to [\alpha]A$, pelo lema A.2, temos que $[\alpha]A \in w$. Assim, como $wR_\alpha w'$, temos que $A \in w'$.

(Provando 2.): Seja $wR_\alpha w'$. Como $\vdash_{\mathcal{LAP}} (\neg Pre_{\mathcal{D}}(\alpha, L) \wedge \neg L) \to [\alpha]\neg L$, então, pelos lemas A.2 e A.3, teremos que $\neg L \in w$ implica $\neg L \in w'$. ■

Finalmente, temos o teorema A.1, que estabelece a completude de $\mathcal{LAPD}$:

Teorema A.1 Se uma fórmula A é $\mathcal{LAPD}$-válida, então ela é um $\mathcal{LAPD}$-teorema.
Prova: Prova-se facilmente pela contrapositiva: se uma fórmula A é $\mathcal{LAPD}$-consistente, então ela é $\mathcal{LAPD}$-satisfatível.

Se A é $\mathcal{LAPD}$-consistente, o conjunto $\{A\}$ também o é. Assim, pelo lema A.1, este conjunto pode ser estendido para um conjunto maximal-consistente w. Claramente, $A \in w$, e pelo lema A.4, temos que $\models^\mu_w A$. Como μ é um $\mathcal{LAPD}$-modelo (pelo lema A.5), temos que A é $\mathcal{LAPD}$-satisfatível. ■

Apêndice B

Tradução entre $\mathcal{LAPD}$ e $\mathcal{LAP}_{\leadsto}$

Apresentamos aqui a equivalência sintática e de expressividade entre $\mathcal{LAPD}$ e $\mathcal{LAP}_{\leadsto}$, definindo traduções de $\mathcal{LAPD}$ em $\mathcal{LAP}_{\leadsto}$ e de $\mathcal{LAP}_{\leadsto}$ em $\mathcal{LAPD}$. Com isso, mostramos que o formalismo de $\mathcal{LAPD}$ abarca todo o poder expressivo do de $\mathcal{LAP}_{\leadsto}$ e vice-versa.

B.1 Tradução de $\mathcal{LAPD}$ em $\mathcal{LAP}_{\leadsto}$

Sejam $BC_{\mathcal{LAPD}}$ e $LEIS_{\mathcal{LAPD}}$ uma base de conhecimento e um conjunto de leis de domínio, respectivamente, em $\mathcal{LAPD}$. Então, $BC_{\mathcal{LAPD}}$ e $LEIS_{\mathcal{LAPD}}$ são traduzidos automaticamente em $BC_{\mathcal{LAP}_{\leadsto}}$ e $LEIS_{\mathcal{LAP}_{\leadsto}}$, com $BC_{\mathcal{LAP}_{\leadsto}}$ e $LEIS_{\mathcal{LAP}_{\leadsto}}$ uma base de conhecimento e um conjunto de leis de domínio em $\mathcal{LAP}_{\leadsto}$. Com isso, definimos o seguinte algoritmo para a geração da relação de dependência binária $\leadsto$ a partir de $\mathcal{D}$:

Algoritmo B.1.1 (Geração da relação de dependência $\leadsto$)

entrada: A relação de dependência ternária $\mathcal{D}$;

saída: A relação de dependência binária $\leadsto$;

Para cada expressão α may cause L if C em $\mathcal{D}$:

Escrever $\alpha \leadsto L$ em $\leadsto$. ■

Como exemplo, a aplicação do algoritmo B.1.1 ao conjunto

$$\mathcal{D} = \left\{ \begin{array}{c} \mathit{carregar} \text{ may cause } \mathit{Carregada} \text{ if } \top \\ \mathit{atirar} \text{ may cause } \neg \mathit{Vivo} \text{ if } \mathit{Carregada} \\ \mathit{atirar} \text{ may cause } \neg \mathit{Carregada} \text{ if } \top \end{array} \right\}$$

produzirá a seguinte relação $\leadsto$:

$$\leadsto = \left\{ \begin{array}{c} carregar \leadsto Carregada \\ atirar \leadsto \neg Vivo \\ atirar \leadsto \neg Carregada \end{array} \right\}$$

Completando a tradução de $\mathcal{LAPD}$ em $\mathcal{LAP}_{\leadsto}$, temos o seguinte algoritmo para produzir os axiomas de persistência condicionais necessários em $\mathcal{LAP}_{\leadsto}$:

Algoritmo B.1.2 (Geração de axiomas de persistência condicionais)

entrada: A relação de dependência ternária $\mathcal{D}$;

saída: Um conjunto de axiomas de persistência condicionais em $\mathcal{LAP}_{\leadsto}$;

Para cada expressão α $\mathsf{may\ cause}$ L if C em $\mathcal{D}$:

Para cada literal L_i, $1 \leq i \leq n$, tal que $C = L_1 \wedge \ldots \wedge L_n$:

Adicionar a $LEIS_{\mathcal{LAP}_{\leadsto}}$:

$\Box((\neg L_i \wedge \neg L) \rightarrow [\alpha]\neg L)$ ■

Como exemplo, a aplicação do algoritmo B.1.2 à expressão

$$atirar\ \mathsf{may\ cause}\ \neg Vivo\ \mathsf{if}\ Carregada$$

produzirá o seguinte axioma de persistência condicional:

$$\Box((\neg Carregada \wedge Vivo) \rightarrow [atirar]Vivo)$$

Juntando os resultados acima, temos, assim, uma maneira de se traduzir representações de domínio em $\mathcal{LAPD}$ para representações em $\mathcal{LAP}_{\leadsto}$.

B.2 Tradução de $\mathcal{LAP}_{\leadsto}$ em $\mathcal{LAPD}$

Seja $BC_{\mathcal{LAP}_{\leadsto}}$ uma base de conhecimento de um dado domínio modelado em $\mathcal{LAP}_{\leadsto}$. Seja também $LEIS^{-}_{\mathcal{LAP}_{\leadsto}} = LEIS_{\mathcal{LAP}_{\leadsto}} - \{\Box((\neg A \wedge L) \rightarrow [\alpha]L); \Box((\neg A \wedge L) \rightarrow [\alpha]L) \in LEIS_{\mathcal{LAP}_{\leadsto}}\}$ o conjunto das respectivas leis de domínio sem os axiomas de persistência condicionais. Então, $BC_{\mathcal{LAP}_{\leadsto}}$ e $LEIS^{-}_{\mathcal{LAP}_{\leadsto}}$ são traduzidos, respectivamente, em $BC_{\mathcal{LAPD}}$ e $LEIS_{\mathcal{LAPD}}$, com $BC_{\mathcal{LAPD}}$ e $LEIS_{\mathcal{LAPD}}$ uma base de conhecimento e um conjunto de leis de domínio em $\mathcal{LAPD}$.

Para gerar a relação de dependência $\mathcal{D}$, definimos o seguinte algoritmo:

Algoritmo B.2.1 (Geração da relação de dependência $\mathcal{D}$)

entrada: A relação de dependência binária $\leadsto$ e o conjunto $LEA_{\mathcal{LAP}_{\leadsto}}$ de leis de efeitos de ações em $\mathcal{LAP}_{\leadsto}$;

saída: A relação de dependência $\mathcal{D}$;

Para todo elemento $\alpha \leadsto L$ em $\leadsto$:

Dados os axiomas de persistência condicionais:

$$\begin{gathered} \Box((\neg A_1 \wedge \neg L) \rightarrow [\alpha]\neg L) \\ \vdots \\ \Box((\neg A_n \wedge \neg L) \rightarrow [\alpha]\neg L) \end{gathered}$$

Acrescentar α $\textsf{may cause}$ L $\textsf{if}$ $A_1 \wedge \ldots \wedge A_n$ em $\mathcal{D}$.

Dada a lei de efeito $\Box(A \rightarrow [\alpha]L)$

Acrescentar α $\textsf{may cause}$ L $\textsf{if}$ A em $\mathcal{D}$. ■

Como exemplo, para a dependência binária

$$\leadsto\; = \left\{ \begin{array}{c} \mathit{atirar} \leadsto \neg\, \mathit{Carregada}, \\ \mathit{atirar} \leadsto \neg\, \mathit{Vivo}, \\ \mathit{carregar} \leadsto \mathit{Carregada} \end{array} \right\}$$

e o conjunto de leis de efeito

$$LEA_{\mathcal{LAP}_{\leadsto}} = \left\{ \begin{array}{l} \Box[\mathit{carregar}]\mathit{Carregada}, \\ \Box[\mathit{atirar}]\neg\, \mathit{Carregada}, \\ \Box(\mathit{Carregada} \rightarrow [\mathit{atirar}]\neg\, \mathit{Vivo}), \\ \Box((\neg\, \mathit{Carregada} \wedge \mathit{Vivo}) \rightarrow [\mathit{atirar}]\mathit{Vivo}) \end{array} \right\}$$

a aplicação do algoritmo B.2.1 produziria o seguinte resultado:

$$\mathcal{D} = \left\{ \begin{array}{c} \mathit{atirar} \textsf{ may cause } \neg\, \mathit{Carregada} \textsf{ if } \top, \\ \mathit{atirar} \textsf{ may cause } \neg\, \mathit{Vivo} \textsf{ if } \mathit{Carregada}, \\ \mathit{carregar} \textsf{ may cause } \mathit{Carregada} \textsf{ if } \top \end{array} \right\}$$

que é o esperado.

Com a tradução de $\mathcal{LAP}_{\leadsto}$ em $\mathcal{LAPD}$, completamos, assim, o estabelecimento da equivalência sintática e de expressividade entre ambos os formalismos.

Índice Remissivo