Universidade Federal do Ceará
Departamento de Computação
Programa de Pós-Graduação em Ciência da Computação

Uma aplicação para a LPR

Autor
Samy Soares Passos de Sá

Orientador
Prof. Dr. Tarcísio Haroldo Cavalcante Pequeno

Co-orientador
Prof. Dr. Marcelino Cavalcante Pequeno

Dissertação de Mestrado apresentada à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal do Ceará como parte dos requisitos para obtençãoo do grau de ***Mestre em Ciência da Computação****.*

Fortaleza – Ceará
Setembro 2008

Capítulo 1

Introdução

"Knowledge is power."ou "Conhecimento é poder.", em português, atestou Sir Francis Bacon (1561 - 1626), em sua obra intitulada em inglês como "Religious Meditations, Of Heresies", de 1597. Em meio à Europa renascentista, com a discussão das origens do homem, da arte, do pensamento, dentre outros em alta, o conhecimento haveria de ser, logicamente, um dos assuntos abordados e tratados com mais fervor. Em uma época de transição onde as armas perdiam força, enquanto ganhavam importância as artes e o conhecimento, o Racionalismo ganhou novo fôlego e tais conceitos foram debatidos em trabalhos diversos. Este momento de transição na história européia, que pode ser visto como um momento intermediário entre a idade média e a moderna, e transição entre o feudalismo e o capitalismo, foi uma época de grande produção intelectual, e com uma série de trabalhos e obras importantíssimos para o desenvolvimento da humanidade enquanto humanidade, restaurou a importância do conhecimento e da transmissão do mesmo. Olhando ainda antes disso em nossa história, podemos dizer que desde o advento da escrita, inicialmente desenvolvida pela necessidade maior das pessoas de se comunicarem, o ato de compartilhar idéias e conhecimento tornou-se possível, bem como seu registro. O passar dos anos e a evolução desta, seja como forma de arte ou ferramenta, possibilitou outros avanços ao homem e a consequente documentação dos mesmos, documentação esta que é a principal origem de muitas afirmações hoje realizadas acerca do passado, e possibilitou às pessoas ensinarem umas às outras, repassar seus ofícios, estratégias para os afazeres do dia a dia e compartilhar e trocas experiências. O conhecimento passou a ter valor, de forma que comunidades diversas tinham cargos e títulos hierarquizados em função da sabedoria demonstrada por seus membros, da capacidade de ensinar e do conhecimento sobre certos assuntos. Hoje mesmo, podemos afirmar que o mesmo ocorre em empresas diversas, e as pessoas de mais alto cargo normalmente são algumas das que melhor conhecem a empresa e seu nixo de atuação. Este conhecimento é colocado como de grande valor dentro dos contextos coerentes ao mesmo e tido como necessário aos representantes destas comunidades, tanto para formar seus sucessores como para levar adiante a própria comunidade. Observamos o mesmo em inúmeros ramos da ciência, e nas diversas relações entre os homens, onde podemos verificar o desejo e necessidade de expressar o conhecimento de forma válida em aplicações e contextos diversos. Queremos registrar novos passos em pesquisas e trabalhos, especificar valores que detalhem o estado atual das finanças em uma empresa para referência futura, anotar uma idéia ou repassar uma mensagem. Todos esses aspectos da vida real envolvem algum tipo de conhecimento, o qual será de nosso interesse empregar, repassar ou guardar com fins tão diversificados quanto as origens destas aplicações, e mais uma vez observamos uma informatização em várias dessas aplicações, o que nos leva a tratar esse aspecto com um pouco mais de detalhe: Uma vez do intuito de repassar e armazenar conhecimento, necessitamos transcrevê-lo ou gravá-lo de alguma forma a preservar o registro e permitir que, tanto em um futuro próximo ou distante, seja possível acessá-lo e recuperar esse conhecimento. Dessa forma, considerando os diversos mecanismos hoje existentes para tal registro, seja uma gravação de voz, com diversas palavras e termos técnicos, ou um enunciado de situação e tabulação de dados numéricos, dentre outros, sempre vamos fazer uso de alguma ferramenta que nos permita expressar o conhecimento e depois recuperá-lo através de uma interpretação do registro. Neste ponto, chegamos ao conceito de informação. Nos cabe pensar em informação simplesmente como uma afirmação ou série de afirmações acerca de algo, sem a necessidade de absorvermos seu conteúdo. Se registramos a latitude da cidade de Fortaleza no mapa geográfico, a informação está ali, independente de quem lê saber o que é latitude e seu significado, embora esses números, para o leigo, não passem de meros dados, estruturas e símbolos os quais não sabemos interpretar; Somente quando tal avaliação dos dados for possível, conseguiremos extrair a informação dali e então poderemos transformá-la em conhecimento, de forma que o 'conhecimento', é uma coleção de informações absolvidas pelo contexto, envolvendo algum tipo de intencionalidade, provenham essas informações de qualquer tipo de dados, sejam registrados ou através de percepção dos sentidos, e o equivalente aos sensores em um agente de inteligência artificial.

É interessante ressaltar com mais detalhe essa natureza dual da conversão que se dá entre informação e conhecimento. Detemos conhecimentos que desejamos registrar, então o fazemos sob a forma de informações. Em seguida, essa informações podem ser obtidas por outros, que podem absorvêlas e tranformá-las em conhecimentos. O passo intermediário, do registro do conhecimento sob a forma de informações, especifica em si mesmo uma característica essencial: A informação precisa existir de forma que possa ser codificada e depois reinterpretada. Precisamos de um elo que permita que as diferentes pessoas, que compartilham seu conhecimento e com quem o conhecimento é compartilhado, tenham, também conhecimento prévio de uma linguagem, seja falada, de símbolos ou gestos, para poder se comunicar. As palavras que compõem essa língua falada, bem como os símbolos e gestos envolvidas em outros estilos de linguagens, são apenas dados. Quando reunidos para que tenham sentido, se tornam informações e, finalmente, quando interpretados e absorvidos, se tornam conhecimento. Para uma pessoa que não detém o conhecimento sobre como decodificar um texto escrito em russo, por exemplo, o mesmo não passa de um emaranhado de símbolos confusos e sem sentido; São apenas dados. Esses dados são extremamente necessários como instrumento para a passagem do conhecimento, pois eles vão determinar o mecanismo de codificação e decodificação, vão ser uma representação, sob a forma de símbolos, do conhecimento que o autor deste texto detinha previamente à sua escrita.

Nesse sentido, encontramos formas de expressar idéias através símbolos em alguma linguagem específica para este propósito, que nos permitam realizar operações sobre o conhecimento expresso em função de gerir consequências e tirar conclusões daí. Através de modelos matemáticos podemos atingir esse objetivo, especificando aspectos de um contexto qualquer sob a forma de símbolos matemáticos, conjuntos e operações sobre os mesmos, discretização de domínios e faixas de valores, ou mesmo gráficos. Tratado dessa forma, falamos de conhecimento simbólico, como as formas de representar conhecimento através de sistemas formais, bem como o próprio conhecimento representado. Um mecanismo particularmente interessante de se utilizar nesse sentido é a lógica matemática, que oferece uma ampla base de resultados e operações, além de sua proposta de permitir representar exatamente as formas de pensamento, deduções, conclusões e trabalhar sobre características dos elementos de algum domínio de aplicação.

Através do uso de símbolos da lógica para representar as informações contidas em uma base de conhecimento, podemos especificar um modelo matemático, lógico, sobre o qual é possível realizar todo tipo de inferências fornecidas pelo formalismo escolhido, dentre os vários trabalhos existentes em lógica e expressividade matemática. Esses formalismos se baseiam em operadores e em alguma linguagem utilizada para especificar um tipo de lógica, em geral dotada de operações específicas que permitam trabalhar as informações de acordo com o propósito próprio para o qual cada lógica tenha sido desenvolvida. Várias propostas diferentes existem, até mesmo porque, a lógica matemática data do tempo dos grandes filósofos gregos, como Aristóteles, e desde então se desenvolvem buscando representar novos aspectos do pensamento humano.

A tarefa de representar conhecimento, seja por lógica ou por quaisquer outros mecanismos, é uma prática comum e necessária em vários contextos. Todo tipo de informação que inserimos em um computador é trabalhada e armazenada sob a forma de símbolos, no caso, zeros e uns, de forma que nos é permitido trabalhar com essas informações de acordo com a abstração realizada em um modelo matemático. É o que ocorre com as representações de caracteres em código binário. Ao escrevermos um texto, trabalhamos as sequências de palavras e caracteres, mas o computador armazena tudo sob a forma de sequências de bits, e toda operação sobre os caracteres do textos é, na verdade, uma operação sobre esses bits. Particularmente, em aplicações de computador que envolvam inteligência artificial, fazemos amplo uso da representação de conhecimento, seja para especificar ambientes, considerar o conhecimento de um agente de IA, ou permitir ao agente avaliar uma situação de forma automatizada. Como exemplo, consideremos um agente autônomo capaz de jogar jogo-da-velha. Este jogo consiste em um pequeno tabuleiro, de nove espaços, em que dois jogadores se revezam em suas ações de preenchimento dos espaços, um por jogada, com o objetivo de desenhar uma linha, em qualquer direção, de quadrados preenchidos com seu símbolo de escolha. Em geral, usa-se $\times$ e $\circ$ como símbolos. O jogo tem um tabuleiro e regras muito simples, e uma rápida análise matemática mostra que é possível criar um autômato que jogue escolhendo sempre a melhor jogada, e garantir que este autômato jamais perderá uma partida, mesmo que não comece jogando. O contexto desse autômato, que envolve sucessivas tomadas de decisão baseadas nas ações do outro jogador, compõe um contexto de agente de inteligência artificial, e o conjunto de regras de transição do autômato representa uma estratégia, especificada em um modelo matemático desse tipo. No exemplo do jogo-da-velha, o raciocínio desempenhado é padronizado, de forma que, baseado em condições específicas, não varia. Esse tipo de aplicação permite uma modelagem, também, por lógica, que neste caso pode-se utilizar inclusive a lógica clássica proposicional, pois a não variação de consequências e previsibilidade do sistema permite um cálculo monotônico.

A característica de monotonicidade, para a matemática, em termos de funções, significa manter alguma característica de ordem inalterada, como ser sempre crescente ou decrescente, mantendo-se fiel a essa característica. Quando falamos em monotonicidade lógica, o mesmo se refere a teorias, de forma que nas lógicas monotônicas, as teorias mantém suas características indefinidamente, e qualquer fórmula, uma vez provada, jamais deixa de fazer parte da mesma, mesmo diante da verificação de novos axiomas. Funcionando de forma contrária, em lógicas não-monotônicas encontraremos teorias em que fórmulas uma vez provadas podem ser removidas desta, deixando de fazer parte das mesmas à medida que novos fatos da teoria são verificados, ou novos axiomas são adicionados à base de conhecimento. Dessa forma, problemas que envolvam incertezas e suposições no caminho para formar conclusões encontram uma forte barreira nas lógicas monotônicas com relação à possibilidade de ter suas características representadas matematicamente de forma apropriada. Qualquer tentativa de representar esses problemas com tais recursos tenderia a ser falha ou incompleta de alguma forma. Para estender a capacidade de representar problemas matematicamente, uma série de formalismos foram propostos envolvendo maneiras de utilizar os operadores e a relação de consequência lógica de forma a permitir provas de conclusões não-monotônicas. Esses formalismos permitiriam representar contextos em que suposições fossem feitas e retiradas em caso de surgirem novos fatos que atestassem sua invalidez, ou seja, em que o conjunto de conclusões pudesse ser reduzido quando do acréscimo de novas premissas. É exatamente nesse ponto em que podemos verificar mais facilmente a relação do termo com a matemática, se pensarmos em uma função que, baseada no número de premissas (argumentos), retorna a quantidade de elementos do conjunto de conclusões da teoria. Neste contexto, tal função aplicada sobre uma teoria tratada por lógica monotônica é crescente em todo seu domínio (função monotônica, portanto), possivelmente apresentando algumas regiões planas em seu gráfico, enquanto que a teoria tratada por uma lógica não monotônica teria um gráfico potencialmente cheio de curvas e possivelmente descontínua (função não-monotônica, portanto).

A aplicabilidade para teorias que necessitem de não-monotonicidade para serem tratadas são muitas. Em um contexto em que façamos inferências sobre o futuro, baseado em fatos da atualidade, estatísticas variadas, opiniões e costumes das pessoas, podemos fazer previsões acerca do dia de amanhã. Essas afirmações são tomadas como verdade no momento em que são formuladas, mas somente após a adição de vários outros fatores, que ainda não ocorreram, poderemos saber se tal afirmação permanecerá ou não na teoria. Por exemplo, podemos supor, como tem acontecido todos os dias, que o Sol irá nascer no dia que segue. Qualquer contexto da teoria que dependa do Sol será tido como adequado na previsão para o próximo dia, porém sabemos, de acordo com cientistas, que o Sol não é eterno, e pode ser que em um dia, quando essa afirmação seja realizada, esteja incorreta, provando-se o erro no dia seguinte. Essa é uma generalização indutiva e constitui um tipo de falácia lógica. Somente se tivermos mecanismos para remover esta afirmação da teoria no momento em que descobrirmos que o Sol não nasceu é que podemos dizer que a teoria se mantém consistente.

Suponhamos um contexto em que a seguinte afirmação é válida:

1. $sunshine(d) \rightarrow sunshine(tomorrow(d))$ (Indução)

Suponhamos agora que no dia 12/12/2012 o sol não nascerá e teremos, em uma teoria monotônica a seguinte afirmativa:

2. $\neg sunshine('12/12/2012')$

Porém, ocorre que:

3. $sunshine('11/12/2012')$ (Fato verificado no dia 11/12/2012)
4. $sunshine(tomorrow('11/12/2012'))$ (Modus Ponens com 1)
5. $sunshine('12/12/2012')$ (Composição do predicado $tomorrow(x)$)
6. $\perp$ (Pois sabemos que $\neg sunshine('12/12/2012')$)

O problema acima ocorre devido à natureza não-monotônica da teoria relativa às previsões do futuro, e pede mecanismos adequados para tratá-la e permitir que conclusões possam ser falsificadas diante de nova evidência. Defendemos que assim ocorre com quase todo contexto interessante, uma vez que os contextos monotônicos são completamente previsíveis e, em geral, simples de tratar. No caso das espécies naturais, temos algumas características gerais reconhecidas em uma maioria dos indivíduos, mas com exceções. Para uma teoria acerca de tal domínio que deseje ser condizente com a realidade, é necessário manter abertas possibilidades de aceitar fatos contrários ao que a teoria em si espera. Somente mecanismos com as capacidades de aceitar um erro e, então, retirar sua palavra ou incorporar novo conhecimento baseado em sua experiência serão capazes de lidar com uma série de contextos existentes no mundo real, fato indubitável nos casos de agentes projetados para atuar em ambientes onde não lhe é possibilitado, por limitações temporárias ou definitivas do contexto, obter todas as informações que lhe seriam necessárias para realizar a melhor escolha.

Dentre as possibilidades de ambientes parcialmente observáveis, ressaltamos, inclusive, a ocorrência de vários momentos no dia-a-dia de cada pessoa quando é necessário tomar uma decisão, desde escolher uma laranja no supermercado ou atravessar uma rua até o fechamento de contrato para uma parceria de negócios. Grande parte das atividades realizadas no dia-a-dia das pessoas é de natureza pouco compreensível e cheia de minúncias às quais nem sempre estamos atentos ou somos informados, dificuldades que se aplicam da mesma forma, em geral, para agentes criados para auxiliar em tarefas comuns ou resolver alguns tipos de problemas. Nos basta imaginar a complexidade do ambiente em que consta o trânsito de uma grande cidade e as dificuldades envolvidas na criação de um agente que controle um robô capaz de atravessar ruas. Podemos ainda ressaltar a forma como comumente nos referimos ao pensamento e tomada de decisões como raciocínio lógico e tratamos certas conclusões por deduções, como, por exemplo, em casos de diagnóstico de problemas, seja de saúda, no computador ou outroa máquina, em função de alguns poucos fatos conhecidos, mas insuficientes para uma afirmação certeira. Na verdade, uma vez que por dedução entende-se um processo cognitivo em que as conclusões são consequência direta, estão embutidas e possuem generalidade em nível tal e qual as premissas que as geraram, esses assim ditos "fatos" são, em verdade, "pistas", e o que temos como conclusões fogem de deduções, pois as inferências realizadas nos levam a informações que *não estão* realmente inclusas nessas pistas. As premissas em questão apenas apontavam algum nível de *evidência* que viabilizam tais conclusões. Tomar por verdade absoluta baseado apenas nessas premissas tende a constituir falácia, uma vez que tal verdade não pode ser provada sem que outras afirmações, fatos ou pistas apareçam para ajudar. Nós realizamos inferências desse tipo, não dedutivas, o tempo todo, e ao avaliar novas afirmações, muitas vezes mudamos de idéia, estendemos nossas conclusões, ou restringimos o domínio das mesmas. Dizemos que tais inferências são indutivas, caracterizadas pela capacidade das premissas de *apoiar* às conclusões, apesar de não serem necessariamente suficientes para garantir sua valoração verdade o que, dessa forma, permite ao raciocínio indutivo descrever toda sorte de manifestação cognitiva em que são feitas *generalizações* baseadas em conjunto de instâncias individuais.

No âmbito de avaliar e promover conclusões mutáveis, o raciocínio indutivo, quando realizado, pode permitir todo tipo de reviravolta de opinião e análise de várias diferentes possibilidades para o mesmo contexto dentre outras formas de conclusões ampliativas com relação à base de conhecimento. Representar esse tipo de inferência com a propriedade adequada à modelagem de tantos tipos de raciocínio ampliativo quanto existem é uma tarefa complicada, pois existem muitos aspectos a serem observados que somente se corretamente tratados poderão representar esse conhecimento e processar tais informações de forma adequada. Ainda assim, nos vale frisar que sem os múltiplos aspectos que tornam tal modalidade de raciocínio difícil de tratar, o mesmo se resumiria à extensão somente do conjunto das proposições conhecidas que derivam das verdades iniciais. Algo que tornaria o ato de gerar conclusões em um contexto realmente mais interessante é exatamente a capacidade de ampliar o conjunto das verdades. A partir de então, se executarmos raciocínio puramente dedutivo, ainda teremos todas as verdades de antes, mas agora com uma maior gama de proposições e informações tidas como de nosso interesse, regentes do mundo ao qual se refere a teoria presente.

Os formalismos especificados para tratar deste tipo de proposições e inferências se refere ao conjunto das lógicas não-monotônicas, frameworks para o trabalho de proposições que não seguem à mesma característica que as lógicas monotônicas de garantir que o conjunto de proposições cresce diante da adição de uma nova cláusula, uma vez que para formalismos não-monotônicos, o conjunto de proposições pode inclusive diminuir de tamanho em tais casos [1]. Como contextos diversos detém essa característica de, por vezes, diante de uma nova informação, ter comprovada a falsidade de algumas suposições, e desde que as lógicas e o raciocínio não-monotônicos têm sido bem aceitos como teorias adequadas ao trato destes tipos de contextos, não resta dúvida de que as práticas dedutivas são limitadas e portanto insuficientes para lidar com certos tipos de raciocínio. Embora contrariadas por linhas mais tradicionais de estudos em representação do conhecimento e do que é ou como se desenvolve o raciocínio, nos gostaria muito poder dizer que as abordagens não-monotônicas são amplamente aceitas. Uma vez que essas correntes tradicionais, ainda não convencidas pelo potencial do indutivo, coloquem em dúvida o valor matemático de conclusões embasadas, porém não garantidas, reconhecem como argumentação correta somente o que é desenvolvido por meios dedutivos, os trabalhos neste ramo de pesquisa com lógicas e raciocínio não-monotônicos foi e têm sido vistos com preconceito e sido descreditados. Ainda assim, quando tratando de assuntos da vida real, em que decisões precisam ser tomadas e nem sempre todas as informações necessárias para uma decisão acertada estão disponíveis, o que ocorre vai além do dedutivo. Em casos como este, ocorre de realizarmos inferências de caráter ampliativo baseado em opiniões gerais, previsões relacionadas e consenso, buscando uma melhor solução que, por vezes, nem mesmo pode ser posta em prática.

Em [20], é posta a seguinte questão: Suponha que estou caminhando pelos *Champs Elysées* e de repente vejo um velho amigo do outro lado da rua. Neste momento, não estou preocupado com horários, pois não tenho nenhum compromisso, e não há tráfego significativo na rua. Após observar com atenção o passar dos carros para escolher um momento apropriado para minha travessia, começo a cruzar a rua. Eis que, enquanto isso, a 10000 metros de altura, a porta do compartimento de carga de um avião se solta e antes de chegar ao outro lado da rua, sou atingido. Teria, nesse caso, sido algo irracional atravessar a rua? Os autores usam ainda de um toque de humor e dizem ser improvável que a manchete dos jornais fosse "Idiota morre ao tentar cruzar rua". O ponto é que alguns fatores nem sempre são conhecidos, e quando raciocinamos, tais fatores acabam sendo desconsiderados. O que devemos tomar por *racional*, deve ser a argumentação com intuito de maximizar o desempenho esperado em uma ação, ou a partir de uma decisão tomada com embasamento nas proposições que compõem a base de conhecimento. Saber o resultado de realizar cada possível ação escolhida coloca a racionalidade em outro patamar, uma vez que podemos fazer uma análise qualitativa dos possíveis resultados e escolher um caminho muito mais difícil de executar, o que o afasta do esperado e racional, quando em função do melhor resultado. Ou ainda, podemos falar em uma espécie de onisciência, uma vez que saber o resultado final de cada ação é muito próximo do caso em que temos todas as informações necessárias para suportar uma decisão nesse contexto, pois esperamos que, se todas as informações estão disponíveis, possa-se perceber os possíveis resultados e escolher a melhor opção no mesmo, desde o princípio.

O tipo de inferência realizada no exemplo acima exemplifica uma ocasião em que a ação escolhida leva a consequências diferentes do esperado. Neste caso, não foi possível completar o objetivo de atravessar a rua, pois houve uma interrupção no meio do percurso. Esse tipo de "erro" com frequência ocorre, simplesmente, porque tais inferências são baseadas em uma série de outras informações não explicitadas, mas implícitas no contexto avaliada. Além disso, muitas vezes, pecamos por generalizações e exageros, entre outras falácias do discurso, muito embora realizar essas suposições possa ser considerado racional, como ocorre com a suposição de *mundo fechado* nesses contextos. Essa suposição rege que, dada uma base de informações sobre um determinado domínio de conhecimento, devemos supor que ali estão todas as informações relevantes, e o que quer que não esteja listado, deve ser considerado alheio a essa contexto. Por exemplo, ao ver listados os nomes de dez objetos, supomos que todos os objetos são diferentes, e dependendo do caso, que aqueles são todos os objetos que deveriam estar listados. Imaginemos, mais especificamente, que em uma nota no cinema esteja escrito que os filmes "Cantando na Chuva", "E o vento levou", "Casablanca" e "A noviça rebelde" estarão em cartaz na semana seguinte. Se perguntados sobre quantos filmes estarão em cartaz na semana seguinte, uma pessoa tende a responder que serão quatro filmes, baseado na suposição de mundo fechado, em que essas seriam todas as informações relevantes. Além disso, uma outra suposição é realizada em função de escolher a resposta quatro neste caso, que é a chamada *Hipótese do Nome Único*, em que os objetos, sendo detalhados por diferentes nomes, diferem uns dos outros. Essas suposições são comumente utilizadas no discurso para inferir sobre quaisquer proposições, mesmo relativas a acontecimentos no dia-a-dia.

Nos aproximando dos contextos de prática computacional, uma das primeiras origens de exemplos trabalhados que mostravam tais limitações, aplicações de bancos de dados pressupunham um problema quanto a relação de suas informações e o mini-mundo modelado pelo mesmo. Perguntava-se, então, se era o caso que, quando, ao acessar o banco, se o retorno da aplicação fosse a inexistência de resposta ou objetos que atendessem à consulta realizada, poderia-se realmente afirmar, com toda a propriedade inerente de quem tem certeza absoluta, que tal resposta ou objeto não existe no mini-mundo. O motivo da pergunta envolve tanto a possibilidade de erro humano ao inserir os dados no banco, quanto a possibilidade de dados terem sido perdidos, dentre outros que poderiam ter levado o banco a ser, em verdade, inconsistente com esse mini-mundo objeto. Embutidas na afirmação do usuário do banco de que tal resposta ou objetivo não existe, estão implícitas ambas as noções de mundo fechado e nome único. Supõe-se a partir destas que os dados ali representam, realmente, ao máximo do que se conhece e é permitido saber naquele momento sobre o mundo modelado. Em outras palavras, espera-se que, sendo o banco consistente com o que se conhece, o contexto representado esteja *completo*, e é a partir dessa suposição que afirmamos realmente não existir resposta para a consulta realizada.

O próprio *PROLOG*, que é uma linguagem de programação interpretada, amplamente adotada pelos programadores do paradigma lógico, e que trabalha com modelos teóricos descritos sob a forma de relações e predicados, utiliza-se dessa suposição de mundo fechado para efeito de seus cálculos, o que afeta os resultados de prova quando utilizamos um operador especial de *negação por falha*. Sendo assumido que todas as informações relevantes estão na teoria e, portanto, que o provador tem acesso a todas essas informações, algumas consultas podem não retornar resultado ou retornar algum resultado incompleto, caso alguma informação esteja omitida, pois a parte não descrita da teoria será tomada como dispensável, falsa, sem precedentes.

Em lógica, o que observamos, bem como em matemática de forma geral, é que ocorre um tratamento diferente deste em seus modelos, no sentido de não considerar, a priori, nenhuma das duas suposições, pois um mesmo objeto pode ser referenciado com diversos nomes, e para ressaltar a possível propriedade de um conjunto de ter n elementos distintos, é necessário especificar através de formulação matemática que os objetos existem em tal número e que são diferentes dois a dois. Fora isso, em raciocínio matemático, a menos que especificado um limite mínimo e máximo para a cardinalidade do conjunto dos filmes disponíveis na próxima semana, deduzimos, simplesmente que algum número entre 1 e infinito de filmes estará disponibilizado. As hipóteses de mundo fechado e nome único não existem, a menos que detalhadas como axiomas da teoria. Compactuando então do princípio da suposição de mundo fechado e informações variadas que definem padrões de formas existenciais, comportamentos observados e expectativas, fazemos tais inferências, às vezes errôneas, o que não ocorre em lógica dedutiva, pois cada aspecto e detalhe da teoria deve estar especificado para gerar novo conhecimento. Essas inferências são muito importantes exatamente porque nem sempre é possível especificar cada aspecto e detalhe da teoria ou faltam informações, e, quando necessária uma conclusão, decisão ou opinião sobre o assunto apontado, para que possamos dizer que se trata de uma conclusão lógica ou racional, baseamo-nos ao máximo no conhecimento presente, mas tendemos a realizar algum tipo de indução sobre estas. Por outro lado, as conclusões que encontramos através da argumentação indutiva envolvem suposições que serão potencialmente colocadas à prova, e uma proposição que negue qualquer dessas conclusões, sendo passível de ser também concluída, caracteriza tais conclusões e o processo argumentativo que as geraram como sendo passíveis de falha ou *derrotáveis*. Essas novas proposições, inicialmente tomadas como possíveis, prováveis ou viáveis verdades, podem ser mais tarde derrubadas ou derrotadas (daí o nome) por novas afirmações adicionadas à base de conhecimento e suas consequências dedutivas. Assim, como principal consequência da adição de expressividade possibilitada por uma relação de conclusão não-monotônica, abre-se espaço para o surgimento de conflitos nas teorias tratadas desta forma. Tais conflitos são gerenciados pelas diversas abordagens de forma a serem resolvidos e permitir à teoria continuar consistente, de acordo com o desejado em todo caso de teoria.

Destes conflitos, o primeiro tipo que encontramos é relacionado diretamente à capacidade derrotável das proposições que especificam conclusões. Uma vez da adição de um fato ou proposição especificada na base de conhecimento e tomada como verdade absoluta no contexto trabalhado, se este fato leva a uma inconsistência quando em conjunto com as porposições geradas por inferências de caráter indutivo, tais inferências são removidas, como que dando-se um passo de volta, pois seu grau de confidencialidade é, em muito, inferior ao grau de confidencialidade da proposição assumida na base de conhecimento. Esses conflitos, que aparecem entre uma proposição derrotável e a base de conhecimento, são, em geral, tratados desta forma pelas abordagens não-monotônicas.

O segundo caso, porém, envolve um aspecto mais complexo da representação de conhecimento e inferências lógicas, e recebe tratamento ligeiramente mais variado. Por conta disso, essa forma de lidar com tais conflitos tem um papel bem mais importante quando da caracterização desses métodos. Este tipo de conflito é observado quando diferentes proposições derrotáveis são incompatíveis, e se faz necessária ou uma forma de resolver entre essas proposições qual deve ser assumida, ou a possibilidade do método de se abster com relação à escolha de alguma proposição. Podemos ir além e dizer que alguma abordagem há de permitir que todas as proposições sejam tomadas como verdades, observando-se diferentes modelos que satisfaçam a cada uma e propondo que todas sejam verdades, dessa forma, cada qual em seu modelo. Enquanto uma abordagem é mais conservadora e prefere se proteger da possibilidade de assumir algo potencialmente errado, dessa vez não apenas sem garantias, mas também com uma evidência, de grau de confidencialidade potencialmente tal e qual o seu próprio, de que possa estar errada, a outra abordagem é mais expansiva e audaciosa, por assim dizer, e assume alguma possibilidade, mesmo em casos em que sua própria negação é uma outra proposição derrotável. Em nossa abordagem, detalhada no capítulo 2, optamos pela opção mais audaciosa, como veremos. Uma vez que podemos também abordar de forma quantitativa e relacionar números às proposições listadas para gerar probabilidades de ocorrência, vale ressaltar que ambas as opções aqui citadas são listadas por um ponto de vista puramente qualitativo e que o mesmo ocorre com a LPR, importando muito mais a pureza de idéias e a representação do conhecimento em si do que mecanismos de decisão para estes casos. O trabalho aqui apresentado utiliza um mecanismo a partir do qual podemos aproveitar conceitos de teoria dos jogos para *quantificar* esses graus de qualidade das conclusões e assim fazer escolha por uma decisão que julguemos ser mais acertada que as outras, uma vez que o contexto da aplicação da LPR que descrevemos em nosso trabalho demanda uma decisão, criando para tal uma *função de utilidade* para avaliar essas possibilidades. Veremos que tal prática é desnecessária nesta aplicação, pois os vários graus apresentados sugerem uma hierarquia com relação à utilidade de cada possibilidade.

Comentando ainda uma última possibilidade para a combinação de proposições incompatíveis, nos restaria citar os casos em que as duas proposições conflitantes pertencem à base de conhecimento, o que corresponde claramente a uma teoria incorreta, com falhas em sua construção e que, portanto, não é de nosso interesse. Casos desse tipo são teorias que permitem provas de absurdos, e a partir daí, de qualquer proposição que seja. Teorias que contém absurdos não são interessante de ser tratadas, pois não há limites para o que pode ser defendido ali, e por isso, não podemos dizer que há debate ou mesmo raciocínio sobre tais contextos.

A seguir, para melhor nos situar na questão de proposições derrotáveis, como exemplo, citaremos um assunto amplamente discutido e polêmico, mas bastante próximo de todos nós, que se mostra particularmente interessante de avaliarmos. A discussão sobre a existência ou não de vida em outro planeta, analisada rapidamente por um ponto de vista simples e formal da lógica, aparece a partir da defesa de duas afirmações diretamente opostas: "Existe vida inteligente em outros planetas" e "Não existe vida inteligente em outros planetas". Como cada um dos lados, ao se expressar com intuito de defender seu ponto de vista, mostra evidências e fatos que suportam suas teorias, teríamos várias proposições, adequadamente representadas em alguma notação lógica suficiente para tal, que seriam compatíveis com as definições de mundo especificadas para modelar o mundo em que vivemos. Dado que, nessas condições, ambas as conclusões são passíveis de serem inferidas, temos um caso de proposições derrotáveis incompatíveis. Uma pessoa *cética* decidirá por se abster, enquanto outras pessoas, *crédulas* tomarão uma escolha e a assumirão como verdade, talvez, inclusive, passando a defender esse ponto de vista.

1.1 Formalismos não-monotônicos

Para lidar com proposições evidenciadas, mas que podem se provar erradas, formalismos diversos foram introduzidos a partir do fim da década de 70 no ramo das práticas orientadas à realização de inferências não-monotônicas, especificando mecanismos para representar as dúvidas embutidas em uma verdade e outras formas de tornar uma proposição derrotável. Estendendo as lógicas monotônicas ao acrescentar operadores e passos inferenciais ao cálculo de predicados, as lógicas não-monotônicas propunham opções viáveis de representar aspectos do raciocínio matemático clássico que não podiam ser representados pelas lógicas tradicionais. A importância de tais novas práticas se viu necessária quando das tentativas infrutíferas de representar, com lógica de primeira-ordem, as incertezas e limitações possíveis (mas não necessárias) dos contextos-objeto de teorias matemáticas e provas formais. A seguir, visitaremos algumas destas propostas, buscando apresentar a intuição por trás de cada uma.

1.1.1 Circunscrição

Baseado na idéia de minimizar o contexto de cada predicado, no que diz respeito à sua aplicabilidade em objetos alheios ao contexto, assumindo dessa forma que informações especificadas por sentenças externas à teoria devem ser desconsideradas, alguns operadores matemáticos de consequência indutiva foram propostos. Dentre os primeiros formalismos sugeridos para resolver o problema de representar o raciocínio indutivo, encontramos o princípio da *Circunscrição* de McCarthy [15]. Este formalismo deixa claro que, todas as outras coisas sendo iguais, a extensão dos predicados utilizados além do que conhecemos na teoria deve ser mínima, o que nos permite representar de forma eficaz idéias generalizadas, como ao dizermos “Todo pássaro normal voa”. A anormalidade de pássaros deve ser tomada, então, como mínima, e esperamos que, a menos que diante de proposições positivas sobre a anormalidade de um pássaro qualquer, ele seja em verdade normal e, portanto, capaz de voar. McCarthy sugeriu então uma maneira de representar esse tipo de idéia formalmente, utilizando lógica de segunda-ordem para poder, então, quantificar sobre predicados. Essa diferença diante da lógica de primeira ordem permite falarmos em propriedade sobre predicados, propriedades de propriedades ou, ainda, *meta-propriedades*. Ao utilizar notação de segunda-ordem para especificar as sentenças da base de conhecimento, nos é permitido afirmar pontos da teoria tais como “Todo predicado é falso em 0”, que se traduziria como $\forall P \neg P(0)$, ou ainda, é possível afirmar $\forall P(P(a) \leftrightarrow P(b))$, para representar a afirmação “Toda característica que ocorre em 'a', ocorre também em 'b', e vice-versa” ou ainda “Não há como discernir 'a' e 'b' por suas características”.

O formalismo de McCarthy utiliza-se da noção de minimalidade na ocorrência dos predicados e pressupõe uma noção de ordem sobre os mesmos. A ordem proposta para os predicados é análoga à noção de continência entre conjuntos, se observarmos um predicado como o conjunto cujos elementos são aqueles termos os quais, quando assumidos pelas variáveis, o tornem verdadeiro. Desta forma, um predicado é menor que outro de acordo com a sentença de segunda ordem $\forall x(P(x) \rightarrow Q(x))$, abreviada, simplesmente, por $P \leq Q$. Mais especificamente, como é desejado especificar a ocorrência do predicado como sendo a mínima possível tal que este permita satisfazer às sentenças que o referenciam, outros predicados com a mesma posição na ordem especificada podem ocorrer; Somente predicados menores é que não. Na circunscrição de P, especificamos que não existe predicado Q, tal que $(Q \leq P) \wedge \neg(P \leq Q)$, ou seja, tal que $Q < P$. Compondo essas informações, e adicionando a notação $A(P)$ para dizer que o predicado P ocorre na fórmula A, dizemos que $A(P) \wedge \neg\exists Q(A(Q) \wedge Q < P)$. Essa fórmula afirma que P satisfaz A sendo o menor, ou seja, sendo o mínimo predicado capaz de fazê-lo, pois não existe predicado Q menor que P que satisfaça a A.

Abrevia-se, ainda, como $A(P) \mid\!\sim \varphi$, a noção de que A tem por consequência φ, desde que P seja o menor possível e satisfaça a A, ou seja, que as ocorrências do predicado P na teoria, sejam consistentes com a sentença A, e somente sejam consideradas válidas as ocorrências descritas na teoria. Essa é a notação utilizada para representar a circunscrição de cada predicado, e é o elemento principal que permite representar inferências indutivas neste formalismo.

Vale ressaltar a necessidade da notação de segunda ordem para representar tal princípio, uma vez que o mesmo é definido através de caracterização sobre predicados, sendo este aplicado a todo e qualquer predicado da teoria. Em contraste, o que acontece em inúmeras aplicações é que se faz necessário, por vezes, especificar a circunscrição para apenas alguns predicados, em função de limitar o potencial de inferências sobre a base de conhecimento, enquanto deixando livres ou limitando de alguma forma somente a ocorrência ou não dos predicados em termos cujos resultados não estão especificados a priori. Em função disso, outros formalismos foram propostos como extensões deste para aumentar a gama de possibilidades expressivas que a relação de consequência induzida pela circunscrição oferece.

Embora ofereça uma abordagem interessante para o problema da suposição de mundo fechado, a circunscrição de McCarthy possui um problema. Como formalismo para representar teorias e permitir inferências indutivas, o trabalho de McCarthy permite uma enorme expressividade frente às técnicas anteriores de representação de conhecimento, porém, ao utilizar-se incondicionalmente de lógica de segunda ordem para compor sua noção principal, as representações de teorias passam a sofrer com a impossibilidade de serem totalmente axiomatizadas, e, por consequência, não lhe serem permitidas gozar de um cálculo inferencial completo. Ao saltar da primeira para segunda ordem, ganhamos expressividade, mas perdemos a garantia de poder representar a teoria como um todo e listar todas as suas validades.

1.1.2 Redes Semânticas

Para nos permitir propôr ontologias em lógica, em tentativas de recriar domínios do mundo real em aplicações para computadores e para fins de documentação, a abordagem por categorias se fez muito presente. Amplamente aceita como base para diversos formalismos de representação do conhecimento, especificar classes de objetos, agrupados em domínio com predicados relacionados próprios de cada grupo é uma maneira formidável de lidar com diversos contextos observáveis no nosso mundo e em criação. Ao representar essas categorias por especificações baseadas em predicados (que são verdadeiros para elementos detentores de alguma dessas qualidades que detalham tais categorias), podemos modelar vários contextos a partir dessa prática utilizando notação lógica e permitindo realizar inferências sobre os elementos destas. Assim, essas relações estabelecidas entre predicados e objetos são notavelmente uma ferramenta interessante para a prática de representação de conhecimento e formalismos posteriores viriam a adotar esses conceitos-chave para sugerir técnicas mais expressivas.

Uma instância dessas técnicas, baseada no princípio de minimização das exceções de regras gerais, mesmo que ainda não formalizado, foi proposta em 1909 por Charles Peirce, uma especificação de hierarquia entre predicados para realizar a ontologia em lógica através de uma notação de arcos e nós chamada *Grafos Existenciais*. A representação de contextos através de *Redes de Hierarquia Semântica*, *Redes Semânticas*, ou ainda, *Cadeias Semânticas* é uma técnica que permite aos predicados serem especializados em casos particulares de ocorrência e especificar-se também suas exceções, trabalhando para representar o contexto através de uma divisão por categorias. Assim como no modelo de representação do conhecimento por *Frames*, essa prática vai detalhar através de *classes* de objetos, todos os elementos da teoria, permitindo relacionar essas classes, inclusive. Como nos próprios diagramas de classe utilizados amplamente para projeto de sistemas de computadores e ontologias de bancos de dados relacionais, uma vez representando hierarquia de classes em especializações definidas por predicado, as cadeias semânticas permitem modelar a teoria em predicados que representam as características padrão de cada objeto, e as porções que formam cada subclasse da mesma, mais uma vez, através de uma sequência de especificações de predicados em camadas, de forma que cada predicado, atende, também, aos predicados de níveis mais elevados na hierarquia, a menos que especificado em contrário, caso em observamos a representação de uma exceção.

A representação dos conceitos de cadeias semânticas é comumente feita sob a forma de de um grafo orientado, como sugerido por Peirce, livre de ciclos, onde as arestas representam a relação base das generalizações, "é-um". Por exemplo, sabemos que uma pessoa comum tem duas pernas, dois olhos, dois braços, anda, fala, entre outras características. A classe dos objetos do tipo pessoa generaliza as classes de objetos bebê, criança e adulto, pois um bebê "é-um" pessoa, assim como ocorre com crianças e adultos. Embutido neste conceito está a intenção de minimalizar casos de adultos, bebês e crianças que não são pessoas, ou seja, quando falarmos de características de adultos, por exemplo, não devemos supor animais adultos que difiram de humanos a menos que tenhamos ocorrências conhecidas de animais adultos não humanos no contexto. Os predicados representados por nodos no grafo são conectados entre si com setas (arestas orientadas) que indicam o sentido da generalização e setas cortadas para indicar que aquela generalização é errônea. Por exemplo, dizemos que pessoas andam e falam, mas bebês são pessoas que não condizem com essa regra. O diagrama 1.1 ilustra nosso exemplo de forma mais adequada.

Muito embora as cadeias semânticas tenham valor como prática para representar o conhecimento muito maior que como técnica para representar inferências sobre raciocínio derrotável, é viável utilizá-las dessa forma, visto que compactua do princípio de minimalidade de anomalias. Além disso, a classificação por categorias permite uma abordagem muito próxima da lin-

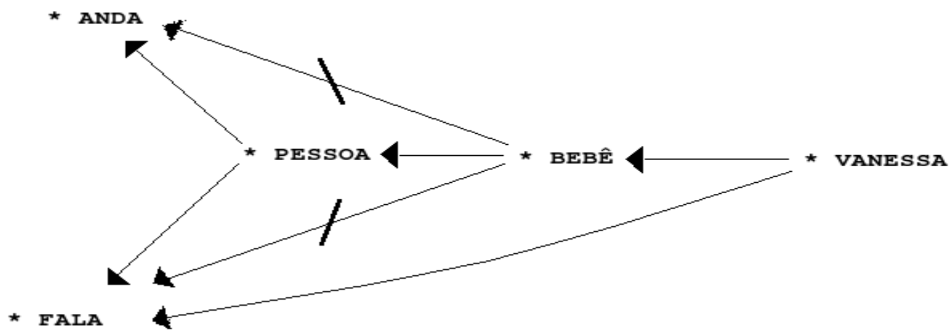

Figura 1.1: Rede semântica PESSOA. Enquanto pessoas em geral andam, bebês em geral não o fazer. Vanessa, porém, é um bebê que anda.

guagem natural e uma fácil visualização da organização dos elementos que compõem a teoria, uma vez que, enquanto os casos mais extremos e particulares se encontram na base da rede, os casos mais amplos e gerais se encontram no extremo oposto. É o que ocorre em nosso exemplo. As características a serem discutidas sobre todos os objetos (Se andam, se falam) estão no nível mais externo, enquanto o caso mais particular, de uma pessoa específica, fica no nível mais interno. A partir do momento que são fornecidas informações sobre um objeto em particular a um sistema baseado em cadeia semântica, as inferências são realizadas do particular para o geral e, a menos que especificadas em contrário (através de novas setas para demonstrar relacionamentos de predicados), assumimos que é verdade a característica mais próxima, sendo essa a detalhada no nodo pai de cada classe. Observamos que cada aresta $A \rightarrow B$ existente afirma, em verdade, que todo A tem, tipicamente e em grande parte, a característica B e que, portanto, a característica A é mais específica. Dessa forma, quando há tal especificação em contrário, verificamos que o modelo segue um princípio de especificidade, sobrepondo à informação padrão, uma mais particular, específica do objeto em evidência. Dizemos que as propriedades listadas valem tipicamente, pois espera-se que objetos compactuem com as propriedades relacionada na rede com o mínimo de excessões não listadas, exatamente como no princípio da circunscrição. A relação entre predicados $A \rightarrow B$ das cadeias semânticas é análoga à relação $A < B$, e as extensões do predicado A nessa relação devem ser mínimas.

Para que as cadeias semânticas possam ser vistas como ferramenta adequada à representação de inferências, faz-se necessário que além de especificar a teoria, seja provido um mecanismo de interpretação para a mesma, assim compondo um formalismo completo. Neste caso, temos abordagens tanto céticas como crédulas, mas em ambos os casos, as inferências são realizadas através do mesmo mecanismo: Definindo-se todas as relações entre predicados através de arestas direcionadas, existirão predicados que não possuem arestas começando em si e outros que não possuem arestas chegando a si. As primeira compõem o nível mais externo da teoria, enquanto os seguintes compõem a parte mais interna. A partir dos predicados de maior especificidade, cujo conjunto consideramos como o primeiro *degrau*, cada novo conjunto será composto pelos predicados que estão a uma mesma distância destes do primeiro degrau. Assim, os vizinhos diretos são o segundo degrau e daí em diante. A partir de então, as inferências são realizadas subindo esses degraus e partindo para asserções mais gerais sobre os objetos trabalhados. Falamos de *extensões* da teoria como sendo conjuntos de novas proposições derivadas a partir da base de conhecimento acerca do contexto exposto, sendo este o termo usual em várias técnicas de inferências lógicas. Na abordagem das redes de hierarquia semântica, as extensões são calculadas subindo os degraus da rede e derivando cada possibilidade na forma de um ramo de árvore com raiz na proposição a ser avaliada e ramificações sempre que houverem opções. No nosso exemplo, teríamos extensões derivadas sobre Vanessa afirmando que ela fala (por termos essa informação aplicada) e também afirmações de que ela não fala (por ser um bebê). Claramente, a noção de degraus só é possível em caso de redes de hierarquia que não possuam ciclos.

Uma peculiaridade desse formalismo que pode ocorrer nessas redes é quando alguma categoria entre as descritas pode ser subconjunto de mais de uma categoria. Em casos como esse, os quais referenciamos como *herança múltipla*, o mecanismo pode derivar mais de um caminho, o que pode tanto ser interessante para a aplicação (em caso de avaliação de cenários), como problemático (como em casos de análise determinística, em que seja necessário garantir as informações). O mesmo ocorre quando um objeto pode pertencer a mais de uma classe diferente e não haja informações sobre qual das classes o mesmo realmente pertence. Asserções múltiplas podem ser feitas nesses casos e o problema se torna multifacetado, de forma que uma abordagem crédula aqui, pode trazer problemas em certas aplicações, como em casos de consultas sobre objetos, e por isso, algumas linguagens de programação orientadas a objetos coíbem modelos com essas características, a exemplo do JAVA.

Ressaltamos ainda que mesmo não proposto como tal, desde que tenham suas semânticas bem definidas, as redes semânticas *são* um tipo de lógica, uma vez que as mesmas permitem um mecanismo de inferência (embutido na relação entre os predicados). Embora sua expressividade seja limitada por girar em torno da relação "é-um", as cadeias semânticas permitem expressar formas de raciocínio indutivo através da ascendência em seus degraus, e a relação induzida diverge em nada da implicação lógica a que estamos habituados, à excessão de incluir a noção de extensão mínima do predicado precedente.

1.1.3 Lógica Default

Os mecanismos que temos discutido apresentam formas de especificar e tirar conclusões de uma teoria baseada em suposições alheias à base de conhecimento. Normalmente, para que esse tipo de conclusão seja aceita em âmbito geral e apoiada por outras pessoas, deve-se expôr algum tipo de explicação, um conjunto de justificativas sobre o assunto, que suportam à decisão e reforçam o sentido de se obter tal conclusão. Baseado neste aspecto das inferências ampliativas, Reiter estendeu as notações então presentes propondo um cálculo sobre fórmulas para representar regras *default*, que apresentavam um novo termo como justificativa para as inferências factíveis sobre as relações, agora entre fórmulas.

Compatativamente, enquanto a circunscrição propõe que predicados na afirmativa que não estejam aplicados a alguns objetos simplesmente não se aplicam a estes, introduzindo a noção de que tais predicados são *tão falsos* quanto lhes for permissível ser, em uma forma particularmente representativa da suposição de mundo fechado, o formalismo introduzido por Reiter permite modelar passos intermediários para a conclusão sob a forma de fórmulas que compactuam com o princípio de circunscrição, relaxando o conceito de minimalização. Isso ocorre, pois as justificativas são suposições tomadas a partir do princípio de minimalização dos predicados, mas os precedentes de uma conclusão são provenientes, também, de uma segunda categoria de requisitos, que não são supostos por circunscrição. As regras de default são especificadas com três componentes, invés de dois, portanto, sendo *Pré-requisito*, *Justificativa* e *Conclusão* esses componentes. Os pré-requisitos precisam ser assumidos verdade antes de realizarmos a inferência, enquanto as justificativas precisam não estarem explicitamente falsificadas no momento da inferência (minimalidade) para que tenhamos a conclusão como resultado. Ressaltamos que as inferências realizadas em cadeia podem concluir fórmulas que serão pré-requisito de outras regras default em momentos posteriores da sequência.

Diferente das técnicas anteriores, as regras default introduzem um *fator* ampliativo, enquanto mantendo o pré-requisito em funcionamento idêntico ao precedente de regras de consequência lógica monotônicas comuns das lógicas dedutivas.

Para representar informações e inferências com essas propriedades, Reiter propôs a seguinte notação para a formulação de regras default:

$\frac{\gamma : \theta}{\tau}$, onde os termos γ, θ e τ são sentenças de uma linguagem e fazem os papéis de pré-requisito, justificativa e conclusão da regra default. Mais especificamente, interpretamos que, uma vez verificado γ, podemos concluir τ, desde que $\neg\theta$ não possa ser verificada verdade.

Essa idéia representada na regra default segue a tendência da idéia de mundo fechado, de forma que prevê limites para as regras em sua justificativa. Repare que a justificativa é o fato de que algum fator na teoria não possa ser provado, o que é verdade caso este fato ou outros que permitissem deduzí-lo não façam parte desta teoria. Como os predicados em teoria default são minimalizados, é exatamente o que se considera, e se a restrição à regra não pode ser provada dentro deste "mundo", temos a justificativa e a possibilidade de utilizar a regra à vontade. Correspondentemente, podemos dizer que a regra default se verifica quando a inferência é aplicável em todos os modelos preferenciais, ou seja, modelos em que ocorrem os menores conjuntos de anomalias às regras gerais, e que, portanto, a negação das justificativas não possam ser provadas, ao passo de que, para uma regra inferencial poder ser utilizada em lógica clássica, é necessário que a inferência seja aplicável em todos os modelos em absoluto, e não apenas em um conjunto de preferenciais.

A capacidade de representar inferências com o grau de generalidade das regras default possibilita modelar conclusões evidentes e sem prova, bem como o processo "dedutivo" que leva às mesmas. As regras iniciais de inferência direta e de default da teoria representam conhecimento certo, e conhecimento evidente, dependendo do tipo de justificativa necessária, e possibilitam modelar certezas absolutas e certezas condicionadas envolvendo o domínio abordado na teoria. A partir de então, através de diretrizes para tratar o conjunto de regras default fornecidas por Reiter, é possível calcular as consequências esperadas da teoria. Assim, inferências ampliativas acerca do mesmo podem ser realizadas dentro do formalismo de lógica dafault, como ocorre em muitos casos de pensamento cotidiano. Esse raciocínio, baseado em justificativas ou situações esperadas, como sugerido nas regras default, quando executado de forma apropriada, produz conclusões sustentadas por argumentos adequados e consistentes com a teoria. Como consequência dessa adição em expressividade, surge uma complicação quanto às conclusões, uma vez que essas regras podem interagir de forma complexa e eliminar às justificativas umas das outras.

Como citado anteriormente, uma teoria default, refenciando-nos dessa forma a teorias que levam regras default em sua composição, permite chegarmos a conclusões sustentadas pela mesma, com justificativas como condições de consistência, mas pode ser dotada de regras entre estas que eliminem a outras regras, modificando a base de conhecimento ao bloquear a possibilidade de realizarmos algumas inferências. Para trabalhar com o domínio de conhecimento de uma teoria de forma apropriada, uma vez descritas as regras default que compõem a mesma, propõe-se o conceito de *extensão* da teoria, como sendo o conjunto das sentenças que podem ser concluídas a partir de um conjunto de defaults Γ. Essa extensão é tal que somente as conclusões que sejam derivadas de forma correta a partir desses defaults pertencem à mesma, e as condições de consistência das inferências devem ser verificadas tanto antes quanto depois da aplicação das regras. Dessa forma, uma vez especificada a extensão da teoria, somente fazem parte desta, regras cujas justificativas não foram comprometidas e podem, portanto, serem aplicadas livremente. Em uma definição circular de extensão de uma teoria default, procuramos um subconjunto maximal de Γ correspondente a todas as conclusões que podem ser tiradas da base de conhecimento, de forma que as regras defaults se aplicam sempre que seus requisitos são alcançados e suas justificativas sejam observadas em uma visão completa do quadro de regras. Reiter utiliza uma noção com um operador recursivo para a construção de um conjunto de sentenças, para, a cada nova aplicação do mesmo sobre a teoria, causar ao conjunto uma evolução ou involução quando da adição de novas regras ao mesmo. Em [1], sugere-se que essa construção é nada mais que um ponto fixo deste operador sobre a teoria, como segue:

Seja uma teoria Default um par (W, Δ), onde W é um conjunto de sentenças e Δ é um conjunto de regras default e considerando que um default pode ser disparado por um par (T_1, T_2) de conjuntos de sentenças, desde que

T_1 prove γ e T_2 *não* prove $\neg\theta$, então dizemos que uma extensão E é construída de forma que:

$E = E_0 \cup E_1 \cup \ldots \cup E_n \cup \ldots,$

onde: $E_0 = W$, e
$E_{n+1} = E_n \cup \{\tau \mid (\gamma : \theta)/\tau \in \Delta$ é disparada por $(E_n, E)\}$

Como definido acima, podemos construir uma extensão de forma que, a cada passo, buscamos pré-requisitos recém alcançados de defaults que ainda não entraram na extensão, e verificamos se o conjunto parcial, candidato a extensão, não produz prova do contrário das justificativas dos mesmos. A partir de então, as conclusões destes defaults são adicionadas à extensão, até que não seja mais possível incluir nenhuma outra sentença, quando chegamos a um ponto fixo da construção e determinamos uma extensão da teoria. Tal construção embute o conceito de circularidade e é utilizada em outros formalismos de raciocínio derrotável para extrair o maior conjunto consistente de sentenças entre si que possa ser montado a partir da teoria, onde se espera que tal conjunto seja, também, o melhor, ou ao menos o mais interessante de se utilizar para realizar inferências sobre a base de conhecimento. Devido à natureza complexa de interação entre as regras default, em algumas teorias podemos encontrar regras que eliminem mutuamente a possibilidade das duas fazerem parte de uma mesma extensão, ou pior, é possível que uma regra especialmente peculiar jamais possa fazer parte de nenhuma extensão, caso sua justificativa seja a própria negação da conclusão. Como consequência, uma teoria default pode gerar várias extensões, ou mesmo não ter nenhuma. Omitiremos exemplos neste momento, pois verificaremos casos em que esses problemas ocorrem mais à frente, quando comentarmos sobre características similares em teorias LPR.

Problemas acerca de teorias default são verificados em ambas abordagens cética ou crédula, pois o cálculo de extensões e a relação entre conclusão e justificativa provocam com que algumas teorias sejam intratáveis em uma ou outra das abordagens. Uma vez que podemos verificar a existência de várias extensões em uma mesma teoria, obtidas pelo método construtivo, de forma que essas extensões são diferentes e não são subconjuntos umas das outras, surge a necessidade de escolher uma destas para ser considerada. Como opções, pode-se testar uma a uma qual aquela que produz resultados mais interessantes à racionalização da teoria, ou simplesmente escolher uma a esmo, o que torna complicado nos apoiarmos na extensão escolhida para defender uma modelagem adequada, a menos que a extensão seja única. Se apostarmos em uma visão crédula sobre os argumentos produzidos utilizando a teoria default, afirmaremos uma sentença qualquer desde que esta seja confirmada e derivada como conclusão em alguma extensão da teoria, o que também apresenta problemas devido ao fato de não podermos determinar algoritmicamente uma extensão preferencial. A abordagem cética, por outro lado, amplamente mais aceita devido ao problema citado, sugere que uma sentença somente seja aceita como produto da teoria se for provada em *todas* as extensões da mesma, e casos problemáticos virão a mostrar que essa abordagem pode não trazer as conclusões adequadas à teoria por verificar inconsistências entre as extensões e eliminar a caracterização monotônica que deve existir sobre W (O conjunto de sentenças da teoria default), mesmo após se determinar uma extensão para produzir inferências.

Citadas algumas opções de abordagens que são oferecidas a um lógico que raciocina sobre algum estilo ou metodologia de trabalho com proposições e conclusões derrotáveis, podemos ponderar sobre outro aspecto importante, do que esperamos ser capazes de representar como raciocínio e comentar sobre a maneira como a LPR, inclusive, lida com este.

1.2 Representação do pensamento em facetas

A forma como as pessoas pensam é tal que aspectos variados, incluindo desde preferências pessoais à inteligência do indivíduo, promove uma enorme variação de conclusões acerca de uma mesma base de conhecimento, variando de pessoa para pessoa. Muito embora o nível de inteligência de um indivíduo possa prejudicar inclusive o senso lógico da decisão, a parte que se refere a preferências e limitações da pessoa se tornam fatos no conjunto de sentenças e podem vir a produzir conclusões diferentes. Quando montamos uma base de conhecimento e definimos as inferências contidas na teoria, porém, especialmente se tratando de pessoas na forma mais geral e abrangente possível, essas preferências precisam ser abafadas ou pré-supostas como o que uma maioria viria a aderir, ou talvez o que manda o bom senso, e tantas outras diretrizes. Supomos sentenças que valham para todo o conjunto, baseado em características gerais, da maioria. Se realizarmos, então, inferências para gerarmos conclusões sobre essa base de conhecimento, encontramos sentenças que esperamos representarem decisões ou conclusões de todos, mas o que realmente acontece é que cada indivíduo, baseado em seus próprios motivos, poderia sugerir um resultado diferente. Um contexto desse tipo nos sugere que, invés de trabalharmos com o todo e para o todo, a forma mais adequada de supor regras gerais para esse conjunto de pessoas é, na verdade, supor várias possibilidades, no caso, sob a forma de tipos de pessoas, categorias, possivelmente baseadas nessas preferências. Da mesma forma, ao raciocinar sobre um problema cujas opções são bem definidas, uma pessoa tende a avaliar cada uma dessas possibilidades, analisando o contexto para exibir preferência sobre uma dessas, seja por questões lógicas pré-supostas no contexto, ou por pura preferência, o que poderia sugerir a composição de uma função de utilidade para tal escolha. Em ambos os casos, verificamos uma natureza multifacetada, com possibilidades e nuances de características entre as mesmas, e um consequente dinamismo na geração de conclusões quando em uma ou outra situações. Esse tipo de situação, envolvendo dinamismo e vários cenários de prova para o mesmo contexto, sugere que o mesmo problema pode se apresentar sob várias facetas, e que o mesmo problema pode apresentar-se aplicado em diferentes circunstâncias, ou em vários diferentes *modelos* da mesma teoria.

Para lidar com essas múltiplas possibilidades, bem conhecido é o formalismo apresentado pelas lógicas modais. Existem várias famílias de lógicas modais, cada qual com mais restrições do que as propostas anteriormente, ou seja, com uma nova propriedade com relação ao comportamento das modalidades presentes em cada sistema. Dentre essas famílias, destacamos a classe *S5*, dotada da características de que qualquer quantidade de modalidades em sequência seja equivalente à circunstância de ter aplicada somente uma ocorrência da última modalidade listada [12], o que nos permite utilizar somente uma ocorrência de modalidade frente a qualquer sentença para expressar sua presença nos modelos da teoria, sem ganhos de expressividade para operadores modais aninhados. Constantemente ligados à lógica modal, encontramos formalismos não-monotônicos que visam representar a natureza do raciocínio ampliativo e que visa avaliar diferentes possibilidades de quadros e situações, representando dessa forma uma maior gama de teorias e práticas inferenciais. Essas lógicas derivadas de formalismos modais e não-monotônicos são referenciadas como pertencendo à família das lógicas *auto-epistêmicas*, da qual a *LPR* também faz parte, ao estender a lógica modal S5 através da inclusão de um conjunto de regras defectíveis (e consequentemente, conjuntos de mundos defectíveis) e o tratamento lógico adequado necessário às mesmas em conjunto com as sentenças monotônicas.

A forma como o caráter ampliativo do raciocínio em LPR é representado envolve uma notação simplificada de inferências derrotáveis da LPR, denominadas simplesmente de *generalizações*, e é uma proposta para modelar regras gerais e inferências indutivas acerca do domínio sobre o qual é proposta a teoria. Utilizamos essa idéia para representar fatos esperados e propôr limites para tal expectativa, de forma que se torna possível ambos hierarquizar as generalizações por conta de seus limites e propôr restrições como a relação "é-um" das redes semânticas, como também é possível produzir extensões a partir de um conjunto de regras desse tipo. Diferentemente da proposta em lógica default, buscamos representar aspectos do raciocínio com nuances diversas tais que a lógica default se mostra inadequada para representá-las, dada a não preferencialidade de extensões geradas, uma vez que a existência de várias extensões acarreta, em verdade, em não determinarmos uma base comum inferencial sobre o problema, e, portanto, tornaria uma discussão sobre o problema infrutífera, incorreta, incompleta ou inapropriada de alguma forma.

Para representar adequadamente à teoria, verifica-se uma enorme importância na existência de *uma única* extensão, de forma que essa sirva de base inferencial comum, sendo consistente e única escolha viável, e teorias formuladas de forma a produzir várias ou nenhuma extensões seriam, em verdade, de alguma forma falhas ou mal formadas, como defendido em [5] e [18].

A seguir, introduziremos a lógica trabalhada em nosso modelo e mostraremos que esta, a *LPR*, proposta sobre um framework modal S5, é uma interessante proposta para representar conhecimento não-monotônico e multifacetado, e que supre vários problemas das técnicas anteriores, propondo uma modelagem eficiente de problemas em lógica, e utilizando, inclusive, uma notação simplificada que consideramos mais intuitiva que a sugerida para especificar defaults.

Capítulo 2

A lógica das Plausibilidades

Para tratar adequadamente da modelagem de contextos em lógica, bem como permitir provas formais dentro de um sistema suficientemente abrangente para compactuar com o que chamamos de raciocínio complexo, buscamos um formalismo que permita avaliar múltiplos cenários simultaneamente, bem como utilizar inferências ampliativas. A LPR é uma proposta de sistema formal que se baseia nas duas características, especificando uma notação própria que permite expressão dos dois aspectos, ou seja, tanto uma forma de especificar inferências derrotáveis, como também novas modalidades são propostas, hierarquizando de forma diversificada a força dos argumentos que compõem as teorias. Essa proposta visa modelar o dito raciocínio *complexo* introduzido anteriormente e fornecer uma forma de realizar prova neste sistema. Para tal, a LPR conta com uma base monotônica, a LPD, que serve para fazer o processamento monotônico dos argumentos que venham a compor uma extensão LPR da teoria, gerando todas as consequências que podem ser monotonicamente derivadas a partir da extensão. Verificaremos que a extensão da lógica modal, bem como todo o tratamento de modelos da teoria se dá diretamente no nível monotônico, e por conta disso, a extensão de teoria proposta faz parte da LPD, especificando uma nova semântica e cálculo axiomático para tratar os múltiplos cenários. Nesta lógica, temos especificadas as modalidades pertinentes, bem como uma definição completa de como são as estruturas modais dessa lógica. Ressaltamos nesse ponto que teorias LPR bem construídas geram somente uma extensão, mas esta especifica vários cenários, onde cada um destes corresponde, em verdade, a alguma extensão no contexto da lógica Default de Reiter. Dessa forma, corrigimos o problema de descentralização das extensões inerentes às teorias default, com relação à decisão de qual tomar por base para cálculos inferenciais.

Em uma teoria LPD, monotônica, sua semântica é definida sobre os conceitos de *estrutura*-LPD e *interpretação*-LPD. Uma *estrutura*-LPD se apresenta como uma dupla de estruturas clássicas propostas sobre o mesmo domínio, de forma que a primeira é a coleção dos mundos possíveis, e a segunda, a coleção dos mundos *plausíveis*. Essas duas estruturas são tais que o conjunto de mundos plausíveis é parte própria do conjunto dos mundos possíveis, o que reflete a característica de maior força argumentativa do plausível sobre o mero possível. Vale ainda ressaltar que a primeira coleção, dos mundos possíveis, é exatamente a apresentada nos formalismos modais tradicionais, e o conjunto dos mundos plausíveis faz parte da extensão proposta para S5. Levando em conta essa noção de estrutura, uma *interpretação*-LPD é associada à mesma, e tal que, dado um mundo extraído do domínio e teoria, a mesma é responsável por um assinalamento das variáveis no domínio comum pré-estabelecido. São apresentadas, ainda, funções associadas a cada interpretação Θ, responsáveis por associar elementos do universo aos termos propostos e estabelecer valorações para cada fórmula na teoria a partir de valores verdade 0 e 1 para falsidade e verdade como definidos em lógica clássica. Os conceitos a seguir são apresentados em acordo com o artigo "A logical expression of reasoning" [5], e a maior parte das definições e exemplos são simplesmente extraídos do trabalho original sobre a *LPR*, cabendo ao autor do presente, somente comentar sobre alguns aspectos deste conteúdo, uma vez do nível de detalhe apresentado no original ser bastante explicativo e didático.

Definição 2.1. *A linguagem utilizada em* LPD *é a linguagem de primeira ordem tradicionalmente definida em diversos trabalhos e livros-texto, utilizando-se de "$\rightarrow$", "$\neg$", "$\Box$" como conectivos primitivos e "$\forall$" como quantificador primitivo por escolha. Adicionamos a esse conjunto um conectivo extra, "!", interpretado como "estritamente plausível", visando ampliar a expressividade do mesmo e permitir-nos estabelecer as fórmulas que se diferenciarão e formarão o conjunto dos mundos plausíveis na estrutura*-LPD.

A notação utilizada, tem como padrões para detalhar as variáveis sintáticas as convenções a seguir:

- L é uma linguagem para a LPD;
- x, y, z são variáveis de qualquer linguagem para a LPD;
- t, u são termos em L;
- P, Q, R, S são fórmulas em L;
- p, q, r são sentenças atômicas de L, de forma que letras distintas representam sentenças distintas.
- Γ, Φ são coleções de fórmulas de L;
- Δ é um conjunto não vazio.

Essas convenções podem ou não ser seguidas por índices numéricos e letras subscritos ou superscritos.

Em função de representar a característica do nosso modelo de raciocínio de relevar múltiplos cenários, especificamos que em uma estrutura-LPD, os termos constantes e funcionais fornecidos pela estrutura devem se apresentar da mesma forma em todos os mundos do domínio, embora os predicados possam variar entre os mesmo. Essa funcionalidade ressalta uma possível variação dos valores verdade de cada predicado aplicado aos vários termos do domínio entre os mundos, de forma a estabelecer a diferença entre os mesmos e ocorrendo por consequência da não-monotonicidade existente na teoria que gera os mundos *plausíveis*. Dizemos que um conjunto de mundos que compactua com essas restrições é *rígido* e definimos como segue:

Definição 2.2. *Um* mundo w sobre Δ para L *é uma função satisfazendo as seguintes condições:*

- *Se c é uma constante em L, $w(c) \in \Delta$;*
- *Se f é uma símbolo funcional n-ário L, $w(f)$ é uma função de Δ^n para Δ;*
- *Se p é uma símbolo predicativo n-ário em L, $w(p)$ é um subconjunto de Δ^n.*

Uma coleção W de mundos sobre Δ para L é dita rígida *se a seguinte condição é alcançada:*

- *Para cada $w, w' \in W$ e para cada s, se s é uma constante ou símbolo funcional em L, $w(s) = w'(s)$.*

Definição 2.3. *As estruturas*-LPD *são tais que provêm um* Universo *e significados para cada termo constante, funcional ou predicativo de L neste universo. Dessa forma, uma* estrutura-LPD *para L é uma tripla $H = \langle \Delta, W, W' \rangle$, onde Δ é um conjunto não-vazio chamado* Universo *de H, e W, W' são coleções* rígidas *não-vazias de mundos sobre Δ para L, tais que $W' \subseteq W$, e cujos elementos são chamados respectivamente de* mundos possíveis *(*in W*) e* mundos plausíveis *(*in W'*) de H.*

Uma *interpretação*-LPD para L é responsável por complementar a estrutura ao prover, além de um universo e significados para os termos constante, funcionas e predicativos, prover um mundo fixo e significados para as variáveis, permitindo assim definirmos o significado de qualquer fórmula ou termo de L, o que se faz essencial para definir propriamente uma semântica para a LPD.

Definição 2.4. *Uma* interpretação-LPD *para L é a quíntupla $\Theta = \langle \Delta, W, W', w, s\rangle$, tal que $H = \langle \Delta, W, W'\rangle$ é uma estrutura*-LPD *para L, $w \in W$ e s é uma função do conjunto de todas as variáveis em L para Δ, também chamada de Δ-*assinalamento *(*de variáveis*). Neste caso, é dito que Θ é uma* interpretação-LPD *para L* sobre *a estrutura*-LPD *H.*

Definição 2.5. *Se s é um Δ-assinalamento para variáveis e $d \in \Delta$, então $s(x|d)$ é o Δ-assinalamento para variáveis definido abaixo:*

- $s(x|d)(y) = \begin{cases} s(y), & \textit{se } y \neq x; \\ d, & \textit{se } y = x. \end{cases}$

Definição 2.6. *Se $\Theta = \langle \Delta, W, W', w, s\rangle$ é uma interpretação*-LPD *para L, $d \in \Delta$ e $w' \in W$, então $\Theta(x|d)$ e $\Theta(w|w')$ são as interpretações*-LPD *para L especificadas abaixo:*

- $\Theta(x|d) \rightleftharpoons \langle \Delta, W, W', w, s(x|d)\rangle$;
- $\Theta(w|w') \rightleftharpoons \langle \Delta, W, W', w', s\rangle$.

Definição 2.7. *Dada uma interpretação*-LPD *$\Theta = \langle \Delta, W, W', w, s\rangle$ para L, as cláusulas a seguir especificam as funções Θ_{D} e Θ_{E}:*

- *Θ_{D} é uma função da coleção de termos em L para Δ, chamada de* denotação para L definida por Θ*;*
- *Θ_{E} é uma função de L para {0,1}, chamada de* avaliação para L definida por Θ*;*
- $\Theta_{\mathrm{D}}(x) = s(x)$;
- *Se c é uma constante em L, $\Theta_{\mathrm{D}}(c) = w(c)$;*
- *Se f é um símbolo funcional n-ário em L,*
  então $\Theta_{\mathrm{D}}(f(t_1, \ldots, t_n)) = w(f)(\Theta_{\mathrm{D}}(t_1), \ldots, \Theta_{\mathrm{D}}(t_n))$;
- *Se p é um predicado n-ário em L,*
  então $\Theta_{\mathrm{E}}(p(t_1, \ldots, t_n)) = 1$ sse $\langle \Theta_{\mathrm{D}}(t_1), \ldots, \Theta_{\mathrm{D}}(t_n)\rangle \in w(p)$;
- $\Theta_{\mathrm{E}}(\neg P) = 1$ *sse* $\Theta_{\mathrm{E}}(P) = 0$;
- $\Theta_{\mathrm{E}}(P \to Q) = 1$ *sse* $\Theta_{\mathrm{E}}(P) = 0$ *ou* $\Theta_{\mathrm{E}}(Q) = 1$;
- $\Theta_{\mathrm{E}}(\forall x\, P) = \min\{\, \Theta(x|d)_{\mathrm{E}}(P) \mid d \in \Delta \,\}$;
- $\Theta_{\mathrm{E}}(\Box P) = \min\{\, \Theta(w|w')_{\mathrm{E}}(P) \mid w' \in W \,\}$;
- $\Theta_{\mathrm{E}}(P!) = \min\{\, \Theta(w|w')_{\mathrm{E}}(P) \mid w' \in W' \,\}$.

*As funções Θ_{D} e Θ_{E} são, portanto, tais que a primeira é responsável por atribuir a cada termo um objeto da interpretação Θ, e a segunda é responsável por determinar a valoração verdade para cada fórmula prevista na estrutura-*LPD.

Definição 2.8. *As estruturas*-LPD H *para* L *especificam funções de* L *para* $\{0, 1\}$*, denotadas por* H_{V}*, e chamada de* valoração-LPD para L definida por H *como segue:*

- $H_{\mathrm{V}}(P) = \min\{\, \Theta_E(P) \mid \Theta$ *é uma interpretação*-LPD *para a linguagem* L *sobre* $H\,\}$.

Definição 2.9. *Seja* H *uma estrutura*-LPD *para* L*, dizemos que:*

- H satisfaz $P \rightleftharpoons H_V(P) = 1$*;*
- H satisfaz $\Gamma \rightleftharpoons H$ *satisfaz cada fórmula em* Γ*;*
- P *é* LPD-satisfatível $\rightleftharpoons$ *existe uma estrutura*-LPD *para* L *que satisfaz* P*;*
- P *é* LPD-insatisfatível $\rightleftharpoons$ *nenhuma estrutura*-LPD *para* L *satisfaz* P*;*
- P *é* LPD-válida $\rightleftharpoons P$ *é satisfeita por toda estrutura*-LPD *para* L.
- Γ *é* LPD-satisfatível $\rightleftharpoons$ *existe uma estrutura*-LPD *para* L *que satisfaz a cada fórmula em* Γ*;*
- Γ *é* LPD-insatisfatível $\rightleftharpoons$ *nenhuma estrutura*-LPD *para* L *satisfaz simultâneamente a todas as fórmulas em* Γ*;*
- Γ *é* LPD-válida $\rightleftharpoons$ *cada fórmula em* Γ *é satisfeita por toda estrutura*-LPD *para* L.

Definição 2.10. *Dizemos que* P *é uma* LPD-consequência semântica *de* Γ *se cada* LPD-*structure for* L *que satisfaz* Γ *também satisfaz* P*. Quando isso acontece, utilizamos a notação* $\Gamma \models_{\mathrm{LPD}} P$ *para representar que* P *é consequência de* Γ.

Definição 2.11. *Diz-se que um* mundo w sobre Δ para L satisfaz P *quando a estrutura*-LPD $H = \langle \Delta, \{w\}, \{w\} \rangle$ *para* L *satisfaz* P*. Similarmente, dizemos que* w satisfaz uma coleção Γ de fórmulas de L *se este mundo satisfaz a cada fórmula da coleção.*

As definições até então apresentadas definem a semântica da LPD como uma *lógica aberta*, o que quer dizer que as regras envolvendo quantificadores universais como generalizações e necessidade gozam de corretude, enquanto que somente uma forma restrita do teorema de dedução vale. Nessas lógicas, se uma implicação segue de uma coleção de fórmulas, então o antecedente desta implica logicamente o consequente sobre essa coleção de fórmulas, mas o contrário dessa regra não vale, ou seja, se uma fórmula P como parte de um conjunto de fórmulas Γ implica logicamente Q, não necessariamente temos que Γ - P implica logicamente a implicação de P em Q. Dessa forma, verificamos a seguinte relação:

Teorema 2.1.

- $\Gamma \models_{\mathrm{LPD}} P \to Q$ *implica que* $\Gamma, P \models_{\mathrm{LPD}} Q$,
  mas $\Gamma, P \models_{\mathrm{LPD}} Q$ *nem sempre implica que* $\Gamma \models_{\mathrm{LPD}} P \to Q$.

Definição 2.12. *Uma* ocorrência de variável x *é dita* ligada *em* P *se ela ocorre dentro de uma subfórmula de* P *da forma* $\forall x\, Q$*. Caso contrário, essa ocorrência é dita* livre *em* P*. Uma* variável *é dita* livre na fórmula *se ocorre livre ao menos uma vez na fórmula. Caso não haja nenhuma ocorrência de variáveis livres em* P*, dizemos que* P *é uma* sentença.

Definição 2.13. *Seja* P_0 *uma sentença de* L *escolhida arbitrariamente, as seguinte convenções são adotadas:*

- $\top \rightleftharpoons P_0 \to P_0$;
- $\bot \rightleftharpoons \neg(P_0 \to P_0)$;
- $P \wedge Q \rightleftharpoons \neg(P \to \neg Q)$;
- $P \vee Q \rightleftharpoons \neg P \to Q$;
- $P \leftrightarrow Q \rightleftharpoons (P \to Q) \wedge (Q \to P)$;
- $\exists x\ P \rightleftharpoons \neg \forall x\ \neg P$;
- $\Diamond P \rightleftharpoons \neg \Box \neg P$;
- $P? \rightleftharpoons \neg\big((\neg P)!\big)$.

Teorema 2.2. *Dada uma interpretação*-LPD $\Theta = \langle \Delta, W, W', w, s\rangle$ *para* L*, a semântica em* LPD *para os conectivos e quantificadores definidos é dada abaixo:*

- $\Theta_{\mathrm{E}}(\top) = 1$;
- $\Theta_{\mathrm{E}}(\bot) = 0$;
- $\Theta_{\mathrm{E}}(P \wedge Q) = \min\{\Theta_{\mathrm{E}}(P), \Theta_{\mathrm{E}}(Q)\}$;
- $\Theta_{\mathrm{E}}(P \vee Q) = \max\{\Theta_{\mathrm{E}}(P), \Theta_{\mathrm{E}}(Q)\}$;
- $\Theta_{\mathrm{E}}(P \leftrightarrow Q) = 1$ *sse* $\Theta_{\mathrm{E}}(P) = \Theta_{\mathrm{E}}(Q)$;
- $\Theta_{\mathrm{E}}(\exists x\ P) = \max\{\Theta(x|d)_{\mathrm{E}}(P) \mid d \in \Delta\}$;
- $\Theta_{\mathrm{E}}(\Diamond P) = \max\{\Theta(w|w')_{\mathrm{E}}(P) \mid w' \in W\}$;
- $\Theta_{\mathrm{E}}(P?) = \max\{\Theta(w|w')_{\mathrm{E}}(P) \mid w' \in W'\}$.

Existe uma hierarquia epistêmica entre as fórmulas da LPD como intencionado, definida de acordo com as especificações a seguir:

Teorema 2.3.

- $P \models_{\overline{\mathrm{LPD}}} \Box P$ *e* $\Box P \models_{\overline{\mathrm{LPD}}} P$*, mas "*$P \to \Box P$*" nem sempre válida na* LPD;
- $\Box P \models_{\overline{\mathrm{LPD}}} P! \models_{\overline{\mathrm{LPD}}} P? \models_{\overline{\mathrm{LPD}}} \Diamond P$.

Acontece, no entanto, que:

Teorema 2.4. *As seguinte proposições nem sempre são verdadeiras:*

- $\Diamond P \models_{\overline{\mathrm{LPD}}} P?$;
- $P? \models_{\overline{\mathrm{LPD}}} P!$;
- $P! \models_{\overline{\mathrm{LPD}}} \Box P$.

LC é uma versão aberta da logica clássica.[1]

O teorema a seguir afirma que a LPD é um extensão conservativa da LC.

Teorema 2.5. *Se Φ e P são respectivamente uma coleção de fórmulas livres de modalidades de L e uma fórmula livre de modalidades de L, então a seguinte proposição é válida:*

- $\Phi \models_{\text{LPD}} P$ *se, e somente se,* $\Phi \models_{\text{LC}} P$.

Lema 2.6. *Seja Γ uma coleção de fórmulas de L de uma das formas Q ou $Q?$, tal que Q é livre de modalidades, de forma que Γ tem ao menos uma fórmula da forma $Q?$. Seja P uma fórmula livre de modalidades de L, e considere $\overline{\Gamma}$ como a coleção $\{\, Q \in \Gamma \mid Q$ é livre de modalidades$\}$. As seguintes proposições são equivalentes:*

1. $\Gamma \models_{\text{LPD}} P?$;
2. *Existe um fórmula* $Q? \in \Gamma$ *tal que* $\overline{\Gamma} \cup \{Q?\} \models_{\text{LPD}} P?$;
3. *Existe um fórmula* $Q? \in \Gamma$ *tal que* $\overline{\Gamma} \cup \{Q\} \models_{\text{LC}} P$.

l Segue a demonstração de que (1) implica (2). Convida-se o leitor a provar as implicações de (2) para (3) e de (3) para (1).

Suponha que $\Gamma \models_{\text{LPD}} P?$.

Existe um subconjunto finito $\Gamma' \cup \{Q_1?, \ldots, Q_n?\}$ de Γ tal que $\Gamma' \subseteq \overline{\Gamma}$ e $\Gamma' \cup \{Q_1?, \ldots, Q_n?\} \models_{\text{LPD}} P?$. Se $n = 0$, então não há nada mais a provar, então consideremos que $n > 0$.

Suponha que, para cada $i \in \{1, \ldots, n\}$, não é o caso de que $\Gamma' \cup \{Q_i?\} \models_{\text{LPD}} P?$. Então, para cada $i \in \{1, \ldots, n\}$, existe uma estrutura-LPD $H_i = \langle \Delta, W_i, W_i' \rangle$ for L tal que H_i satisfaz $\Gamma' \cup \{Q_i?\}$ e H_i não satisfaz $P?$. Se H é uma estrutura-LPD para L definida por $H = \langle \Delta, W_1 \cup \ldots \cup W_n, W_1' \cup \ldots \cup W_n' \rangle$, então H satisfaz $\Gamma' \cup \{Q_1?, \ldots, Q_n?\}$, mas não satisfaz $P?$, o que é absurdo.

Dessa forma, exite um $i \in \{1, \ldots, n\}$ tal que $\Gamma' \cup \{Q_i?\} \models_{\text{LPD}} P?$.

A LPD é uma lógica monotônica elaborada especialmente para trabalhar com raciocínio em cenários múltiplos, de forma que, uma vez dados os cenários possíveis e plausíveis, a estrutura-LPD H os representa nas coleções de mundos W e W', respectivamente. A fórmula plausível $P?$ valer em H significa dizer que existe algum cenário plausível no qual P vale. Já a fórmula estritamente plausível $P!$ valer em H significa que P vale em todos os cenários plausíveis, de forma similar à que $\Diamond P$ e $\Box P$, respectivamente, se aplicam nos cenários possíveis. A seguir, exploraremos o cálculo proposto para a LPD e compreenderemos as principais diferenças no tratamento dos

[1]Versões abertas da lógica clássica são apresentadas em [13, 17, 21], enquanto que versões fechadas da lógica clássica são dadas em [2, 6, 7].

mundos plausíveis e possíveis a partir da apresentação das formas de tirar conclusões através de provas em LPD e da utilização das novas modalidades introduzidas, $\Box$ e $\Diamond$.

2.1 Axiomas da LPD

A seguir, apresentaremos as definições propostas em [5] de cálculo axiomático para a LPD. Essa proposta é feita de acordo com o estilo aberto, ou seja, sem nenhuma restrição para aplicação de regras de inferência para a introdução do quantificador universal $\forall$ ou o conectivo de necessidade $\Box$, mas restringindo a introdução da implicação. Versões abertas da lógica clássica são apresentadas em [13, 17, 21]. Em [3, 4] é fornecido um método geral para a introdução da implicação em formas abertas de cálculo. O método fechado de lidar com regras de inferência de introdução não é utilizado aqui, pois nesse caso, do estilo fechado, apresentado em [2, 6, 7], não é possível derivar "$P/P!$", uma das principais diretrizes que são base para a modelagem em LPD.

Em função de termos uma forma mais fácil de utilizar regras de introdução da implicação, é fornecido ainda um tipo de rastreamento do uso das regras utilizadas para introduzir o quantificador universal e conectivo de necessidade através do uso de entidades chamadas *objetos variantes*, que correspondem a variáveis em lógica sem modalidades. Quando essas modalidades existem, se faz necessário ao menos um outro objeto variante para indicar uma variação entre os mundos. No nosso caso, utilizamos como convenção o sinal "$\Box$". O rastreamento citado então funciona associando a cada regra de inferência aplicada um conjunto de objetos variantes, que pode eventualmente vir a ser vazio.

Definição 2.14. *Um* objeto variante em LPD *é uma variável ou o sinal "$\Box$".*

A definição abaixo, junto à definição 2.12, especifica quando um objeto variante em particular é livre em uma fórmula dentro da LPD.

Definição 2.15. *A fórmula P é dita* $\Box$-fechada *se P tem uma das formas $\Box Q$, $Q!$, $\neg R$, $R \to S$ ou $\forall x\, R$, onde R e S são $\Box$-fechadas; caso contrário, dizemos que $\Box$ é livre em P.*

Definição 2.16. *Dizemos que uma fórmula é* modal *se ocorre algum dos sinais "$\Box$" ou "!"; caso contrário, a fórmula é dita* livre de modalidades *ou* modality-free.

Definição 2.17. *$P(x|t)$ é a fórmula obtida de P através da substituição de t por cada ocorrência livre de x, e repondo ocorrências de variáveis consistentemente ligadas em P por outras que não ocorrem em P quando necessário.*[2]

Definição 2.18. *O cálculo para a* LPD *tem os seguinte postulados (esquemas de axiomas e regras de inferência), nos quais, para cada regra de inferência, um objeto variante pode ser anexado:*

1. $P \to (Q \to P)$;
2. $(P \to Q) \to \Big(\big(P \to (Q \to R)\big) \to (P \to R)\Big)$;
3. $\dfrac{P,\ P \to Q}{Q}$, *onde nenhum objeto variante está anexado;*
4. $(\neg P \to Q) \to \big((\neg P \to \neg Q) \to P\big)$;
5. $\forall x\, P \to P(x|t)$;
6. $\forall x\, (P \to Q) \to (\forall x\, P \to \forall x\, Q)$;
7. $P \to \forall x\, P$, *onde x não é livre em P;*
8. $\dfrac{P}{\forall x\, P}$, *onde x é o objeto variante anexado;*
9. $\Box P \to P$;
10. $\Box(P \to Q) \to (\Box P \to \Box Q)$;
11. $P \to \Box P$, *onde $\Box$ não é livre em P;*
12. $\dfrac{P}{\Box P}$, *onde $\Box$ é o objeto variante anexado;*
13. $\Box P \to P!$;
14. $P! \leftrightarrow P$, *onde $\Box$ não é livre em P;*
15. $(P! \to P)!$;
16. $(P \to Q)! \to (P! \to Q!)$;
17. $\forall x\, (P!) \to (\forall x\, P)!$.

---

[2]Esse tipo de substituição evita que ocorrências de variáveis em t se tornem ligadas $P(x|t)$. Uma definição detalhada de $P(x|t)$, levando em conta que todas as variáveis ocorrendo em t devem permanecer livres em $P(x|t)$, pode ser encontrada em [2, 6].

Definição 2.19. *Como é usual, uma relação de consequência sintática "$\mid_{\overline{\mathrm{LPD}}}$" é definida, relacionando coleções de fórmulas em* LPD *a fórmulas em* LPD*. Além disso, é definido "$\mid^{\mathcal{V}}_{\overline{\mathrm{LPD}}}$", onde $\mathcal{V}$ é uma coleção de objetos variantes:*

- *Uma dedução $\mathcal{D}$ em* LPD depende de *uma coleção $\mathcal{V}$ (de objetos variantes) se $\mathcal{V}$ contém a coleção de objetos variantes o de todas as aplicações de regras em $\mathcal{D}$ contendo uma hipótese em que o é livre, tal que existe uma fórmula, justificada como premissa em $\mathcal{D}$, onde o também é livre, relevante para essa hipótese em $\mathcal{D}$.*
- *P e uma* consequência sintática de Γ na LPD dependendo de $\mathcal{V}$ *se existe uma dedução de P a partir de Γ em* LPD *dependendo de $\mathcal{V}$; utilizamos, para indicar essa dependência, a notação $\Gamma \mid^{\mathcal{V}}_{\overline{\mathrm{LPD}}} P$.*

Definição 2.20. *Uma fórmula P é dita uma* tese da LPD *se* $\mid_{\overline{\mathrm{LPD}}} P$.

Definição 2.21. *A coleção Φ de fórmulas de L é dita* LPD-trivial *se, para cada fórmula P de L, $\Phi \mid_{\overline{\mathrm{LPD}}} P$.*

Teorema 2.7. *Todos os sinais "$\rightarrow$", "$\wedge$", "$\vee$", "$\leftrightarrow$", "$\neg$", "$\forall$" e "$\exists$" se comportam em* LPD *como na lógica clássica aberta*[3]*. Segue abaixo uma formulação para uma versão do teorema da dedução para a* LPD*:*

- *Se* $\begin{cases} \Gamma \cup \{P\} \mid^{\mathcal{V}}_{\overline{\mathrm{LPD}}} Q, \\ \textit{no varying object de } \mathcal{V} \textit{ is free in} \\ P, \end{cases}$ *então* $\Gamma \mid^{\mathcal{V}}_{\overline{\mathrm{LPD}}} P \rightarrow Q$.

Teorema 2.8. *As modalidades aléticas "$\Box$" e "$\Diamond$" se comportam em* LPD *da mesma forma que na versão aberta da lógica modal* S5*, ou seja:*

- $\Box P \mid^{\emptyset}_{\overline{\mathrm{LPD}}} P$*;*
- $P \mid^{\Box}_{\overline{\mathrm{LPD}}} \Box P$*;*
- $P \mid^{\emptyset}_{\overline{\mathrm{LPD}}} \Diamond P$*;*
- *se Q é $\Box$-fechada, então $\Diamond P, P \rightarrow Q \mid^{\Box}_{\overline{\mathrm{LPD}}} Q$.*

Teorema 2.9. *As modalidades não-aléticas "$!$" e "$?$" têm respectivamente as seguintes regras para introdução e eliminação:*

- $P \mid^{\Box}_{\overline{\mathrm{LPD}}} P!$*;*
- *se Q é $\Box$-fechada, então $P?, P \rightarrow Q \mid^{\Box}_{\overline{\mathrm{LPD}}} Q$.*

[3]Como é apresentada, por exemplo, em [13, 17, 21]; Em [3, 4] conceitos gerais sobre o cálculo aberto, objetos variantes e teoremas de dedução são analisados, juntamente com a relação de consequência "$\mid^{\mathcal{V}}$" e outra similar.

Ao acrescentar essas regras para as modalidades de plausibilidade, as caracterizamos sintáticamente de acordo com a idéia anteriormente introduzida de que essas contém parte do significado representativo semântico das modalidades aléticas tradicionais, uma vez que somente uma das relações de P com $\Box P$ vale para $P!$, e da mesma forma para as relações de P e $\Diamond P$ e $P?$.

Teorema 2.10. *Se* LPD$'$ *é um cálculo axiomático obtido do cálculo axiomático para a* LPD *através da adição do esquema de axioma "$P! \to P$", então as seguintes proposições são verdadeiras:*

- $\vdash_{\mathrm{LPD}'} P \to P?$;
- $\vdash_{\mathrm{LPD}'} P! \leftrightarrow \Box P$;
- $\vdash_{\mathrm{LPD}'} P? \leftrightarrow \Diamond P$;
- $\Gamma \vdash^{\mathcal{V}}_{\mathrm{LPD}'} P$ *se, e somente se,* $\Gamma! \vdash^{\mathcal{V}}_{\mathrm{LPD}} P!$, *onde* $\Gamma! = \{\, P! \mid P \in \Gamma \,\}$.

Teorema 2.11. *As seguintes proposições mostram em maior detalhe a interrelação entre os conectivos de necessidade, plausibilidade cética, plausibilidade crédula e possibilidade em* LPD*:*

- $\vdash_{\mathrm{LPD}} \Box P \to P!$;
- "$P!/\Box P$" *não é uma regra válida em* LPD*;*
- $\vdash_{\mathrm{LPD}} P! \to P?$;
- "$P?/P!$" *não é uma regra válida em* LPD*;*
- $\vdash_{\mathrm{LPD}} P? \to \Diamond P$;
- "$\Diamond P/P?$" *não é uma regra válida em* LPD.

As relações de consequência sintática e semântica da LPD são equivalentes.

Teorema 2.12 (Corretude e completude da LPD)**.**

- $\Gamma \vdash_{\mathrm{LPD}} P$ *se, e somente se,* $\Gamma \models_{\mathrm{LPD}} P$.

l Para auxiliar na prova, constrói-se, inicialmente, uma lógica LPD$'$, sem modalidades, com semântica e cálculo próprios e três tipos de quantificadores primitivos.

A linguagem L' de LPD$'$ é construída a partir da linguagem L da LPD de acordo com as seguintes condições:

1. cada constante de L continua uma constante em L';
2. cada símbolo funcional n-ário de L continua um símbolo funcional n-ário em L';
3. todas as variáveis de L continuam variáveis em L', chamadas *variáveis normais de L'*, mas existe mais uma *variável extra v*, variando entre todos os mundos possíveis;

4. para cada predicado p de L, p é um predicado $n+1$-ário de L';
5. somente existem dois conectivos primitivos em L': "$\to$" e "$\neg$";
6. existem três quantificadores primitivos em L': "$\forall$", "$\forall_\Box$" e "$\forall_!$".

Os termos e fórmulas em LPD$'$ são definidos de acordo com as seguintes cláusulas:

1. cada termo de L é um termo de L';
2. a variável adicional v de L' é um termo de L';
3. se p é um predicado n-ário de L e $t_1, \ldots, t_n$ são termos de L, então p é um predicado $n+1$-ário de L', $t_1, \ldots, t_n, v$ são termos de L' e $p(t_1, \ldots, t_n, v)$ é uma fórmula de L';
4. se P, Q são fórmulas de L', então $\neg P$ e $P \to Q$ são fórmulas de L';
5. se x é uma variável normal de L' e P é uma fórmula de L', então $\forall x\, P$ é uma fórmula de L';
6. se P é uma fórmula de L', então $\forall_\Box v\, P$ e $\forall_! v\, P$ são fórmulas de L'.

Um mundo sobre Δ para L' é simplesmente um mundo sobre Δ for L. Uma estrutura-LPD$'$ para L' é simplesmente uma estrutura-LPD for L. Uma interpretação-LPD$'$ Θ para L' é uma quádrupla $\langle \Delta, W, W', s\rangle$, na qual $\langle \Delta, W, W'\rangle$ é uma estrutura-LPD$'$ para L', e s é uma função definida para todas as variáveis de L', tal que s associa cada variável normal a um elemento de Δ, e associa a variável extra v a um elemento de W. Dada uma interpretação-LPD$'$ $\Theta = \langle \Delta, W, W', s\rangle$ para L', as funções Θ_D e Θ_E, chamada respectivamente a *denotação para L' definida por Θ* e *a avaliação para L' definida por Θ*, são especificadas de forma análoga à utilizada na definição 2.7, com as seguintes diferenças:

1. se p é um símbolo predicativo $n+1$-ário em L', e $t_1, \ldots, t_n$ são termos em L (e portanto, também em L'), então $\Theta_\mathrm{E}(p(t_1, \ldots, t_n, v)) = 1$ sse $\langle \Theta_\mathrm{D}(t_1), \ldots, \Theta_\mathrm{D}(t_n)\rangle \in s(v)(p)$;
2. $\Theta_\mathrm{E}(\forall_\Box v\, P) = \min\{\Theta(v|w)_\mathrm{E}(P) \mid w \in W\}$;
3. $\Theta_\mathrm{E}(\forall_! v\, P) = \min\{\Theta(v|w)_\mathrm{E}(P) \mid w \in W'\}$.

Uma LPD$'$-valoração para L' definida por uma estrutura-LPD$'$ H para L' é especificada da mesma forma que foi feita anteriormente definição 2.8, e finalmente, uma semântica para a LPD$'$ é especificada de forma idêntica à que foi feita na definição 2.9.

LPD$'$ é uma lógica de três tipos, na qual, para uma dada estrutura-LPD$'$ $H = \langle \Delta, W, W'\rangle$ para L', os tipos são Δ, W e W', onde uma variável normal varia sobre Δ, enquanto a única variável extra v varia sobre a coleção W de mundos possíveis. O quantificador universal "$\forall_! v$" obriga v a variar somente sobre W', a coleção de mundos plausíveis de H.

Para cada par de linguagens correspondentes L e L' para as lógicas LPD e

LPD$'$, considere f uma função de L para L', definida através das seguintes cláusulas:

1. se p é um predicado n-ário em L, e $t_1, \ldots, t_n$ são termos em L, então $f(p(t_1, \ldots, t_n)) = p(t_1, \ldots, t_n, v)$;
2. $f(\neg P) = \neg f(P)$;
3. $f(P \to Q) = f(P) \to f(Q)$;
4. $f(\forall x\, P) = \forall x\, f(P)$;
5. $f(\Box P) = \forall_{\Box} v\, f(P)$;
6. $f(P!) = \forall_! v\, f(P)$.

Um cálculo para a LPD$'$ é definido simplesmente através da reescrita de todos os postulados do cálculo para a LPD em uma linguagem para a LPD$'$, levando em conta a função de tradução f. Dessa forma, não é difícil provas que, dada uma coleção Γ de fórmulas de L e uma fórmula P de L, se $\Gamma' = \{f(Q) \mid Q \in \Gamma\}$ e $P' = f(P)$, as seguintes proposições são válidas:

- $\Gamma \mathrel{|}\!\!\underline{}_{\text{LPD}}\, P$ se, e somente se, $\Gamma' \mathrel{|}\!\!\underline{}_{\text{LPD}'}\, P'$; (1)
- $\Gamma \models_{\text{LPD}} P$ se, e somente se, $\Gamma' \models_{\text{LPD}'} P'$. (2)

É também fácil provar que o cálculo para a LPD$'$ é correto e completo com respeito à semântica para a LPD$'$, ou seja:

- $\Gamma \mathrel{|}\!\!\underline{}_{\text{LPD}'}\, P$ se, e somente se, $\Gamma \models_{\text{LPD}'} P$. (3)

Das proposições (1), (2) e (3), segue, finalmente, que o cálculo para a LPD é correto e completo com respeito à semântica da LPD:

- $\Gamma \mathrel{|}\!\!\underline{}_{\text{LPD}}\, P$ se, e somente se, $\Gamma \models_{\text{LPD}} P$.

As seguinte proposições afirmam respectivamente versões sintáticas do teorema 2.5 e lema 2.6.

Teorema 2.13. *Se Φ e P são respectivamente uma coleção de fórmulas livres de modalidades de L e uma fórmula livre de modalidades de L, então a seguinte proposição é válida:*

- $\Phi \mathrel{|}\!\!\underline{}_{\text{LPD}}\, P$ *se, e somente se,* $\Phi \mathrel{|}\!\!\underline{}_{\text{LC}}\, P$.

s Segue diretamente dos teoremas 2.5 e 2.21.

Lema 2.14. *Considere Γ como uma coleção de fórmulas de L de uma das formas Q ou $Q?$, tal que Q é livre de modalidades, onde Γ tem ao menos uma fórmula da forma $Q?$. Seja P uma fórmula livre de modalidades de L, e considere ainda $\overline{\Gamma}$, a coleção $\{Q \in \Gamma \mid Q$ é livre de modalidades$\}$. As seguintes proposições são equivalentes:*

1. $\Gamma \mathrel{|}\!\!\underline{}_{\text{LPD}}\, P?$;
2. *existe uma fórmula* $Q? \in \Gamma$ *tal que* $\overline{\Gamma} \cup \{Q?\} \mathrel{|}\!\!\underline{}_{\text{LPD}}\, P?$;
3. *existe uma fórmula* $Q? \in \Gamma$ *tal que* $\overline{\Gamma} \cup \{Q\} \mathrel{|}\!\!\underline{}_{\text{LC}}\, P$.

Segue diretamente do lema 2.6 e teorema 2.21.

Para compreender por completo o funcionamento da LPD em termos de representação cognitiva, resta-nos verificar ainda um último aspecto das estruturas-LPD: de onde vêm esses tão falados cenários? Como fazer a distinção entre plausíveis e possíveis? Temos visto todo o funcionamento sintático e semântico de um instrumento dedutivo para trabalhar *sobre* esses cenários, mas não vimos ainda como encontrá-los. Como dissemos, aLPD é uma lógica dedutiva capaz de prover a capacidade de realizar inferências e compôr deduções a partir dos mundos possíveis e plausíveis fornecidos na estrutura. Essa lógica serve de suporte monotônico para a LPR, no sentido de que ela nos permite definir as teorias abrangidas pelos axiomas que compõem a base do contexto trabalhado. A complexidade de tal trabalho, que demanda a criação desta lógica provém da necessidade de trabalhar múltiplos cenários complementares ou mesmo antagônicos baseados nesse contexto. A partir do momento em que fazemos múltiplas asserções de natureza ampliativa sobre o mesmo, o que resulta é uma série de quadros ou cenários de forma que estes diferem entre si simplesmente pelas considerações que são tidas como verdadeiras ou falsas em cada um. Se passarmos então a analisar cada cenário individualmente e começar a tirar conclusões dedutivas dos mesmos, podemos descobrir o que procuramos, encontrar resultados interessantes ou desinteressantes, mas de uma forma ou de outra, encontraremos resultados diferentes a partir de cada cenário. Entre conclusões e mundos plausíveis e possíveis, a LPD é a ferramenta utilizada para processar simultâneamente a todos esses cenários e permitir verificar a consistência do processo para encontrarmos aquele que contém o máximo de informações e considerações interessantes, como se as idéias fossem prontamente reunidas de forma conveniente em um mesmo cenário. Este ponto é crucial para o entendimento do relacionamento da LPD com a LPR. Como defendido em [5], as formas de raciocínio científico, cotidiano, de âmbito profissional e investigativo são tais que envolvem em muitos momentos a característica ampliativa e asserções baseadas em intuição, hipóteses a serem testadas, imaginação, e muitas vezes são observadas em quantidade, em múltiplos cenários, múltiplas suposições, etc, de forma que as duas características representam um papel bastante importante no caminho para representar o raciocínio em si, e o processo inferencial pelo pelo qual se passa para realizá-lo e chegar às conclusões apontadas. Como é de nosso interesse representá-lo nessa forma *complexa*, ou seja, multi-facetada e ampliativa, utilizamos a LPR e a LPD para surprir-nos dos recursos necessários. A primeira é responsável por produzir uma estrutura-LPD apropriada, gerando os conjuntos de mundos plausíveis e possíveis, e consequentemente, os vários cenários que representam diferentes pontos de vista do mesmo problema ou contexto, enquanto que a segunda permite tirar conclusões de cada cenário desses, dedutivamente.

Em função de caracterizar propriamente esses conjuntos, considera-se dois conjuntos de premissas, *hard* e *soft*. Esses conjuntos caracterizam uma diferenciação entre aquelas que são consideradas verdades incondicionais, ou ao menos supostas como tal para servir de base para a argumentação, e as asserções baseadas em evidência, e aquelas que são levadas em conta baseado em evidências e hipóteses criadas. Dessa forma, as premissas *hard* são essenciais para determinar quais fórmulas são ou não consistentes com o problema desde o início, limitando o escopo do mesmo, enquanto que as premissas *soft* são as premissas dotadas de algum grau de certeza, mas que não são garantidas, introduzindo dessa forma a característica ampliativa à teoria, tanto quando participando em passos na dedução de conclusões como na sua própria participação no conjunto de premissas. Dessa forma, o conjunto de premissas *soft* pode ser por si só inconsistente caso relevado classicamente, podendo inclusive conter regras contraditórias, o que nos provoca, intuitivamente a considerar esses cenários múltiplos, ambos como plausíveis, e investigar as conclusões derivadas por cada qual.

O que ocorre a partir dessas definições, portanto, é que utilizando-nos das premissas *hard* em conjunto com quaisquer proposições consistentes com estas, podemos gerar mundos possíveis, e temos como mundo possível qualquer modelo suportado por estas. Dizemos que essas proposições, por serem consistentes com as premissas *hard*, são, da mesma forma que os mundos derivados a partir dessas, possíveis. Similarmente, construímos, a partir das premissas *soft* os mundos que chamamos de plausíveis, baseados em conjecturas e proposições plausíveis. Dizemos que um conjunto maximal de conjecturas consistentes entre si é um cenário plausível. O mundo plausível, então, nada mais é do que um mundo que é possível, ou seja, satisfaz às premissas *hard* e, também, a um cenário plausível. Essa característica reflete uma força argumentativa adicional dos mundos plausíveis sobre os possíveis, uma vez que suas regras são consistentes com outras que se espera serem verdades. Em caso de conjecturas opostas, discordantes ou que, em um mesmo conjunto, o tornem inconsistente, separamos as mesmas em diferentes cenários plausíveis, o que resulta, por sua vez, em diferentes mundos plausíveis, e reflete a característica complexa do pensamento como entendemos. A LPR consegue reunir todos esses cenários em uma mesma estrutura e cálculo relacionados de forma a permitir uma análise qualitativa dos argumentos e permitir concluirmos novas proposições que tenham sentido a partir desta. Essa análise é feita relevando o quão confiável cada proposição é com base na verificação de seu valor verdade em cada conjunto de mundos, de forma que uma proposição é dita *fracamente plausível*, ou simplesmente *plausível*, se é provada em ao menos um cenário plausível, enquanto que dizemos que uma proposição é *estritamente plausível*, ou ainda, *fortemente plausível*, se é provada em todos os cenários plausíveis. Uma asserção é plausível, então, não apenas porque é consistente com as premissas *hard*, mas porque em adição a isso, é suportadas por um conjunto de conjecturas que as evidenciam.

A seguir, apresentamos os detalhes técnicos acerca das idéias apresentadas e sua formulação matemática.

As premissas, formando o que dizemos ser uma base LPR, onde (LPR significa *Logic of Plausible Reasoning*), são apresentadas inicialmente em dois conjuntos distintos. O primeiro é um conjunto de fórmulas de primeira ordem, representando as *premissas* hard, e o segundo é o conjunto das *premissas* soft, fórmulas anotadas sob a forma de *generalizações*, uma notação estendida elaborada para representar as conjecturas. Generalizações são uma proposta para tal que os autores consideram bastante flexíveis, uma vez que as mesmas permitem representar condições para sua validade, limitando o escopo de dedução que envolve essas conjecturas através da especificação de exceções para conjecturas que tenham tais supostos limites.

As premissas de uma base LPR são trabalhadas em função de detectar a agrupar as generalizações compatíveis em conjuntos consistentes das mesmas. O que nos possibilita trabalhar as generalizações dessa forma é o conceito de *extensão* a ser construída a partir das informações da base como sugerida na definição 2.32. Conjecturas compatíveis, extraídas das generalizações e sempre sinalizadas com um sinal ? para indicar que essas são fórmulas plausíveis são adicionadas à extensão em função de torná-la maximal em termos do número de fórmulas da base, estado em que está retratado o melhor nível de conhecimento possível sobre a teoria modelada para cada cenário, ou conjunto de conjecturas consideradas, restringindo ao máximo quais são os mundos plausíveis e, consequentemente, as possibilidades levadas em conta no raciocínio desenvolvido.

Um *cenário plausível* (definição 2.38) é um conjunto de fórmulas livres de modalidades geradas pelas premissas *hard* juntamente com as conjecturas extraídas de um conjunto maximal de generalizações compatíveis e fechado para a dedução. Sentenças estritamente plausíveis ($P!$) são aquelas derivadas das generalizações, e que pertencem a todos os cenários plausíveis (a definição 2.37 de generalização fortemente engatilhada explica essa funcionalidade). *Mundos plausíveis* são, portanto, os modelos clássicos dos cenários plausíveis. Entretanto, sentenças plausíveis são válidas em alguns cenários plausíveis e portanto verdadeiras em alguns mundos plausíveis, enquanto que sentenças estritamente plausíveis são válidas em todos os cenários plausíveis e, portanto, verdadeiras em todos os mundos plausíveis.

Ressaltamos que modelos para as premissas *hard* resultam nos mundos possíveis, e portanto qualquer mundo plausível é também um mundo possível. Sentenças possíveis ($\Diamond P$) são aquelas consistentes com as premissas *hard* e são válidas em ao menos um, potencialmente vários cenários possíveis (assim como vale para mundos possíveis). Sentenças necessárias ($\Box P$) são consequências lógicas das premissas *hard*. Elas valem em todos os cenários possíveis (e mundos). O conjunto de teoremas (sentenças que podem ser provadas) de uma teoria LPR é o fechamento dedutivo na LPD da teoria formada pelas premissas *hard*, as sentenças possíveis, e as sentenças plausíveis e estritamente plausíveis.

Dessa forma, quando todo esse tratamento lógico é realizado com a base LPR inicial, nós chegamos a uma teoria que descreve múltiplos cenários que podem ser tratados, tanto semântica como sintáticamente, na lógica LPD para a dedução plausível apresentada nesta seção. A teoria na LPD formada a partir da base LPR é mostrada na definição 2.39, e a estrutura-LPD que a satisfaz é dada na definição 2.41.

A seguir, apresentamos as definições necessárias para a compreensão que desejamos da LPR e os conceitos citados previamente:

Definição 2.22. *Uma* generalização *(*em *L) é uma expressão da forma "$P \prec Q$" tal que P, Q são fórmulas livres de modalidades (de L)*[4]*; em tal expressão, P é chamada* a conjectura *e Q* a restrição *ou* exceção *dessa generalização. Uma* instância da generalização *$P \prec Q$ (*em *L) é uma expressão $P' \prec Q'$, onde P', Q' são* instâncias consistentes de *P, Q (*em *L)*[5]*.*

O sentido objetivado da generalização "$P \prec Q$" é que P representa uma conjectura que é válida sob a condição de que a restrição Q não o seja. Lemos "geralmente P, a não ser Q".

Definição 2.23. $P \prec \rightleftharpoons P \prec \bot$.

Definição 2.24. *Uma base* LPR *(em L) é um par $\tau = \langle T, G\rangle$, onde T é uma coleção de fórmulas livres de modalidades (de L) e G é uma coleção de generalizações (em L).*[6] *T e G representam respectivamente as premissas* hard *and* soft *da base* LPR *τ.*

[4]A notação estabelecida em 2, pág. 29, continua valendo. Uma formula livre de modalidades é definida em 2.16, pág. 35 e significa que uma fórmula não contém os símbolos "$\Box$" ou "!".

[5]Ou seja, variáveis ocorrendo tanto em P e Q são substituídas pelos mesmos termos (em L).

[6]L é uma linguagem fixada para a LPD cujo alfabeto contém todas as constantes, símbolos funcionais e predicados que ocorrem tanto em T quanto em G, e em todas as possíveis conclusões que se deseje extrair de uma base LPR $\langle T, G\rangle$.

Definição 2.25. *Uma coleção de instâncias de generalizações de G em L é dita uma* regra em τ. *Uma coleção finita de instâncias de generalizações de G em L é dita uma* regra finita em τ.

De agora em diante, em todo o texto, a menos que dito em contrário, consideraremos:

- $\tau = \langle T, G\rangle$ é uma base LPR em L;
- a letra G seguida por naturais e/ou subscritos denota uma regra em τ.

Definição 2.26. *Dada uma regra G' em τ, ela é especificada como:*

- $\mathrm{Conj}(G')$[7] $\rightleftharpoons \{P \mid$ *existe um Q tal que "$P \mathrel{-\!\!(} Q$" pertence a G'*$\}$;
- $\mathrm{Rest}(G')$[8] $\rightleftharpoons \{Q \mid$ *existe um P tal que "$P \mathrel{-\!\!(} Q$" pertence a G'*$\}$.

Definição 2.27. *Se P é uma formula e $x_1, \ldots, x_n$ são as variáveis livres em P, então:*

- $\mathrm{uc}(P)$, o fechamento universal de P, *é a fórmula* $\forall x_1 \ldots \forall x_n\, P$;
- $\mathrm{ec}(P)$, o fechamento existencial de P, *é a fórmula* $\exists x_1 \ldots \exists x_n\, P$.

Definição 2.28. *Se $\Phi = \{P_1, \ldots, P_n\}$ é uma coleção finita de fórmulas de L, então:*

- $\bigwedge \Phi \rightleftharpoons \mathrm{uc}(P_1 \wedge \ldots \wedge P_n)$;
- $\bigvee \Phi \rightleftharpoons \mathrm{ec}(P_1 \vee \ldots \vee P_n)$.

As seguintes proposições são válidas:

- $\bigwedge \emptyset = \top$;
- $\bigvee \emptyset = \bot$.

Lema 2.15. *Seja Φ uma coleção finita de fórmulas. As seguintes proposições são válidas:*

- $\Gamma \cup \bigwedge \Phi \vdash_{\mathrm{LPD}} P$ *se, e somente se,* $\Gamma \cup \Phi \vdash_{\mathrm{LPD}} P$;
- $\Gamma \cup \bigwedge \Phi \vdash_{\mathrm{CL}} P$ *se, e somente se,* $\Gamma \cup \Phi \vdash_{\mathrm{CL}} P$.[9]

Dentro de uma base LPR $\tau = \langle T, G\rangle$, uma forma objetiva de ler as generalizações, levando em conta a função que cada uma individualmente tem para o significado do conjunto, uma vez que o comportamento de cada uma delas depende de si mesma e de todas as outras. Para cada regra finita G'

[7] Lê-se $\mathrm{Conj}(G')$" como "*conjecturas de G'*.

[8] Lê-se $\mathrm{Rest}(G')$" como "*restrições de G'*.

[9] Para a segunda proposição, nós também consideramos que Γ e Φ são coleções de fórmulas livres de modalidades, e que P é uma fórmula livre de modalidades.

em τ, G' pode ser lida como "$\bigwedge \mathrm{Conj}(G')$ é uma conjectura em τ, a menos que $\bigvee \mathrm{Rest}(G')$ seja verdade".

Informações que venham das regras são marcadas por uma "?". A partir do ponto em que uma regra pode ser uma coleção infinita de instâncias de generalizações, as informações embutidas nas suas conjecturas e nas restrições às quais a mesma está submetida são extraídas utilizando sub-regras finitas. Nós as chamamos respectivamente de *hipóteses* e *limites* da regra, como segue:.

Definição 2.29.

- $\mathrm{Hyp}(G')$[10] $\rightleftharpoons \{(\bigwedge \mathrm{Conj}(G''))? \mid G''$ *é finita e* $G'' \subseteq G'\}$;
- $\mathrm{Lim}(G')$[11] $\rightleftharpoons \{(\bigvee \mathrm{Rest}(G''))? \mid G''$ *é finita e* $G'' \subseteq G'\}$.

As seguintes proposições são válidas:

- $\mathrm{Hyp}(\emptyset) = \{\top?\}$;
- $\mathrm{Lim}(\emptyset) = \{\bot?\}$.

A seguir, definimos o conceito de uma *extensão em uma* base LPR τ. Extensões são conjuntos de fórmulas fechados para a dedução em LPD que complementam o conhecimento garantido com conjecturas extraídas das generalizações. As generalizações são avaliadas como um conjunto interdependente com o objetivo de verificar se cada uma deve ou não ser incluída na extensão. A inclusão de uma generalização depende da teoria completa, ou seja, proposições derivadas das premissas *hard* e generalizações, uma vez que precisa ser verificado se a sua restrição pode ou não ser derivada na extensão. Mais que isso, essas generalizações não são tomadas uma a uma, mas em conjuntos de instâncias de generalizações, ou regras como definidas em 2.25, com duas condições de restrições de integridade: Suas hipóteses devem ser consistentes com as premissas *hard* em T, e seus limites precisam ser não deriváveis na extensão. Em termos mais técnicos, dizemos que, para cada regra G' em τ, $\mathrm{Hyp}(G')$ é incluída em uma extensão se nenhum de seus limites é provado na mesma. A construção de ponto fixo na definição 2.32 reflete o caráter não local das extensões[12] Neste trabalho, não abordaremos essas noções alternativas, uma vez que o foco do trabalho é a produção de uma aplicação-exemplo de utilização desta lógica para modelagem do conhecimento e processo de prova sobre o mesmo.

---

[10] "Lê-se $\mathrm{Hyp}(G')$" como "*hipóteses de G'*".

[11] "Lê-se $\mathrm{Lim}(G')$" como "*limites de G'*".

[12] O termo "extensão" e a definição através de pontos fixos segue da idéia geral sugerida no artigo original por Reiter sobre lógica default [19]. No trabalho original que introduz a LPR, os autores oferecem outras formas equivalentes de construir as extensões que não apelam para ponto-fixo.

Definição 2.30.

- $\mathrm{Th}_{\mathrm{LPD}}(\Phi) \rightleftharpoons \{P \mid P$ *é uma fórmula de* L *e* $\Phi \mid\!\overline{}_{\mathrm{LPD}}\, P\}$*;*
- $\mathrm{Th}_{\mathrm{CL}}(\Phi) \rightleftharpoons \{P \mid P$ *é uma fórmula livre de modalidades de* L *e* $\Phi \mid\!\overline{}_{\mathrm{CL}}\, P\}$*, onde* Φ *é uma coleção de fórmulas livres de modalidades de* L*.*

Definição 2.31. $\Psi_\tau(\Phi)$ *e* $\overline{\Psi}_\tau(\Phi)$ *são respectivamente as menores coleções de fórmulas de* L *e de regras em* τ *que satisfaçam as seguintes condições:*

1. $T \subseteq \Psi_\tau(\Phi)$*;*
2. *se* $\Psi_\tau(\Phi) \mid\!\overline{}_{\mathrm{LPD}}\, P$*, então* $P \in \Psi_\tau(\Phi)$*;*
3. *se* $\mathrm{Lim}(G') \cap \mathrm{Th}_{\mathrm{LPD}}(\Phi \cup \mathrm{Hyp}(G')) = \emptyset$*, então* $\mathrm{Hyp}(G') \subseteq \Psi_\tau(\Phi)$ *e* $G' \in \overline{\Psi}_\tau(\Phi)$*.*

Repare que a condição (3) verifica se é o caso de que os limites da regra não são provados em LPD a partir de Φ e suas próprias hipóteses.

$\overline{\Psi}_\tau(\Phi) = \{G' \mid \mathrm{Lim}(G') \cap \mathrm{Th}_{\mathrm{LPD}}(\Phi \cup \mathrm{Hyp}(G')) = \emptyset\}$.

Definição 2.32. *Uma coleção* E *de fórmulas de* L *é dita uma* extensão em τ *se* $\Psi_\tau(E) = E$*; neste caso a coleção* $\overline{\Psi}_\tau(E)$ *é denominada o* conjunto de regras geradoras de E em τ.

Essa definição aponta a extensão como o menor ponto-fixo do operador $\Psi_\tau(\Phi)$, de forma que decidimos sobre a participação ou não de generalizações no terceiro item do processo, e as avaliamos por instâncias, averiguando as possibilidades de prova dos limites de cada instância de regra antes de permitir a entrada de suas hipóteses. As condições para entrada de uma instância qualquer na extensão são especificada exatamente em função de verificar se ocorre de a teoria, junto às hipóteses da regra provarem ou não seus limites. Essa definição circular em que o cálculo que indica a entrada ou não de uma regra G_0 na extensão depende da própria regra impede automaticamente um dos problemas que temos com as extensões de Reiter, pois regras do tipo $P(x)$ -($P(x)$ são descartadas pela extensão *LPR* e não se apresentarão em nenhum cenário. Além disso, ciclos pares geram aqui dois cenários em uma mesma extensão, sem trazer problemas à solução, como ocorre com as extensões default. Interessante ressaltar, ainda, que o menor ponto-fixo alcançado por este operador equivale ao conjunto *maximal* de generalizações de G, ou seja, àquele que contém o maior número possível de instâncias de generalizações em cada cenário observado.

Seja

$$\begin{aligned} &T = \emptyset, \\ &G = \{\, \mathrm{flies}(x) \prec \mathrm{penguin}(x)\,,\ \mathrm{feathered}(x) \prec \mathrm{chick}(x)\,\}. \end{aligned}$$

T e G formam uma base elementar LPR que trata de pássaros. Como a partir de τ não pode ser provado que exista algum pinguim ou que exista algum pinto, nós temos que

$$E = \mathrm{Th}_{\mathrm{LPD}}\Bigg(\Big\{ \Big(\forall x \left(\mathrm{flies}(x) \wedge \mathrm{feathered}(x)\right)\Big)? \Big\}\Bigg)$$

é a única extensão em τ.

Considere outra base LPR que é quase igual à base descrita acima, mas com

$$T = \{ \mathrm{penguin}(\mathrm{Tweety}), \mathrm{chick}(\mathrm{Woody}) \},$$

onde G continua como no exemplo anterior. Agora, como pode ser prova do de T que Tweety é um pinguim, e que Woody é um pinto, as instâncias

$$\mathrm{flies}(\mathrm{Tweety}) \multimapinv \mathrm{penguin}(\mathrm{Tweety})$$
$$\text{e}$$
$$\mathrm{feathered}(\mathrm{Woody}) \multimapinv \mathrm{chick}(\mathrm{Woody})$$

ficam bloqueadas, enquanto que as instâncias

$$\mathrm{flies}(\mathrm{Woody}) \multimapinv \mathrm{penguin}(\mathrm{Woody})$$
$$\text{e}$$
$$\mathrm{feathered}(\mathrm{Tweety}) \multimapinv \mathrm{chick}(\mathrm{Tweety})$$

continuam podendo ser aplicadas, então a única extensão em τ é igual a

$$\mathrm{Th}_{\mathrm{LPD}}\Big(T \cup \{ \left(\mathrm{flies}(\mathrm{Woody}) \wedge \mathrm{feathered}(\mathrm{Tweety})\right)? \}\Big).$$

Uma extensão em τ pode mudar de acordo com a linguagem considerada. Por exemplo, considere uma base LPR onde

$$T = \{\mathrm{penguin}(\mathrm{Tweety})\},$$
$$G = \{\mathrm{flies}(x) \multimapinv \mathrm{penguin}(x)\}.$$

Se L é uma linguagem formada com os símbolos não lógicos que aparecem na base (com tem sido implicitamente assumido nos exemplos acima), então $E = \mathrm{Th}_{\mathrm{LPD}}(T)$ é a única extensão. Entretanto, considerando a linguagem de τ como uma linguagem que inclui um símbolo funcional unário "f" e infinitos símbolos constantes ("Tweety" e, para cada $i \geq 1$, "c_i"), além dos símbolos predicativos "voa" e "pinguim". Agora, a extensão em τ é dada por

$$E = \mathrm{Th}_{\mathrm{LPD}}\left(T \cup \left\{ \left(\bigwedge\{\mathrm{flies}(t_1), \ldots, \mathrm{flies}(t_n)\}\right)? \;\middle|\; n \geq 1\right\}\right),$$

onde cada t_i é qualquer termo distinto de "Tweety" e de cada variável. Os termos t_i não podem ser variáveis porque as fórmulas adicionadas a E são fechadas universalmente, e isso viria a significar que todo indivíduo voa, o que não é verdade, uma vez que não nos é permitido inferir que Tweety voa (Até onde vai a teoria do exemplo, também não nos é permitido inferir que Tweety não voa). As duas asserções "Tweety voa" e "Tweety não voa" são possíveis, mas não plausíveis. Na lógica LPR, como veremos na definição 2.39, "$\Diamond$ flies(Tweety)" e "$\Diamond\neg$ flies(Tweety)" são teoremas da base τ, mas "flies(Tweety)?" e "$\neg$ flies(Tweety)?", não.

Os seguintes teoremas afirmam alguns resultados triviais sobre os casos limite.

Teorema 2.16. *As seguintes proposições são equivalentes:*

- $\overline{\Psi}_\tau(\Phi) \neq \emptyset$;
- $\emptyset \in \overline{\Psi}_\tau(\Phi)$;
- Φ *não é* LPD-*trivial.*

Teorema 2.17.

- *Se* $\overline{\Psi}_\tau(\Phi) = \emptyset$ *ou* $\overline{\Psi}\tau(\Phi) = \{\emptyset\}$, *então* $\Psi_\tau(\Phi) = \mathrm{Th}_{\mathrm{LPD}}(T)$.

Teorema 2.18. *As seguintes proposições são equivalentes:*

- $\Psi_\tau(\Phi) = \mathrm{Th}_{\mathrm{LPD}}(T)$;
- *para cada* $G' \in \overline{\Psi}_\tau(\Phi)$, $\mathrm{Hyp}(G') \subseteq \mathrm{Th}_{\mathrm{LPD}}(T)$;
- $\mathrm{Hyp}(\overline{\Psi}_\tau(\Phi)) \subseteq \mathrm{Th}_{\mathrm{LPD}}(T)$.

Definição 2.33. *Dizemos que* G_1 é subjulgada por G_2 *se existe* G_0 *tal que* $G_0 \subseteq G_2$ *e* G_1 *é obtida de* G_0 *através da instanciação simultânea, feita de forma consistente*[13], *de variáveis livres por termos em* L. *Quando* G_1 *é subjulgada por* G_2, *utilizamos a notação* $G_1 \preceq G_2$, *e dizemos que* G_1 *é uma* sub-regra de G_2. G_1 *é dita* equivalente a G_2, *e utilizamos a notação* $G' \approx G''$, *se* $G_1 \preceq G_2$ *e* $G_2 \preceq G_1$.

Definição 2.34. *Seja* E *uma extensão em* τ. *Dizemos que* G' *está* dentro de E em τ *se* $G' \in \overline{\Psi}_\tau(E)$. *Se existir uma extensão* E *em* τ *tal que* G' *está dentro de* E *em* τ, *dizemos que* G' *é* compatível em τ, *ou que* G' *é uma* coleção de generalizações (mutuamente) compatíveis em τ. *Dizemos que* G' *e* G'' *são* co-extensionais em τ *se se ambas estiverem dentro da mesma extensão em* τ. *Se* G' *e* G'' *são co-extensionais em* τ, *mas* $G' \cup G''$ *não está dentro de nenhuma extensão em* τ, *então dizemos que* G' entra em conflito com G'' em τ, *ou que* G' *e* G'' *são* conflitantes em τ, *ou ainda, que algumas*

[13]Ou seja, ocorrências de uma mesma variável, mesmo ocorrendo em generalizações distintas, devem ser substituídas pelos mesmos termos.

generalizações dentro de G' conflitam com *algumas generalizações dentro de G''* em τ.

- Se E é uma extensão em τ, então $\overline{\Psi}_\tau(E) = \{\, G' \mid G'$ está dentro de E em $\tau\,\}$.

Lema 2.19. *As seguintes proposições são válidas:*
- *se $G' \in \overline{\Psi}_\tau(\Phi)$ e $G'' \preceq G'$, então $G'' \in \overline{\Psi}_\tau(\Phi)$;*
- *se G' está dentro de uma extensão E em τ, então cada sub-regra de G' também está dentro de E em τ.*

Definição 2.35. *Seja E uma extensão em τ de forma que G' está dentro de E em τ. G' é* maximal *dentro de E em τ se, para toda G'' dentro de E em τ tal que $G' \preceq G''$, $G' \approx G''$.*

2.1.1 Teorias bem formadas

A forma como a LPR trata as teorias, incluindo cláusulas derrotáveis e premissas incontestáveis, é tal que o produto do tratamento lógico proposto por ela é uma única extensão para teorias que dizemos serem bem formadas. Através da modalidade de plausibilidade fraca, podemos sugerir uma extensão única que permite tanto a uma fórmula e sua negação coexistirem, desde que acompanhadas por esse símbolo, no sentido de que ambas as possibilidades podem vir a ser verdade, e são relacionadas a evidências de que o são. Dessa forma, uma teoria elaborada adequadamente permite somente uma extensão em LPR, maximal, como vimos, e se, em contrário, tivermos uma teoria que não deriva extensão ou permite várias através dessa construção, sugere-se que essa teoria seja mal formada ou ainda *defeituosa*. Quando generalizações diferentes permitem derivar exceções de outras generalizações, ou ainda, quando uma generalização permite provar a exceção de si própria, dizemos que uma teoria que tenha conjecturas extraídas de generalizações desse tipo entre seus elementos contém um *ciclo*, o que gera o problema. No caso de uma generalização que permite provar sua própria exceção, temos um ciclo auto-derrotável, em no caso de generalizações que, quando juntas, provam suas próprias exceções, temos um ciclo mútuo. Em [5], os autores ponderam ainda sobre uma sutileza envolvida nas exceções das generalizações: o fato que as mesmas falam indiretamente das generalizações, apontando seus limites e falando sobre como pode ou não ser usada a informação das mesmas, o que implica que é indevido, em uma generalização, que ela leve à prova de sua própria exceção.

Na condição de proposta para resolver o problemas de modelagem de teorias derrotáveis através das generalizações, a LPR mantém uma relação hierárquica entre suas regras subentendida em seus cálculos, de forma que, uma vez que a regra precisa submeter-se à condição de não derivação de sua exceção para que sua conjectura se possa aplicar, se faz necessário que tenhamos, de antemão, conhecimento sobre essa prova. Só então podemos complementar a extensão com adição de uma nova afirmativa em virtude da consideração dessa conjectura. Nota-se que, pela construção sugerida, a primeira regra, e a ordem em que outras são inseridas na extensão indicam traços de uma hierarquia entre essas regras, uma vez que conjecturas de uma podem concluir a exceção de outras. Em contrário a muitos outros formalismos anteriores, a LPR pondera as exceções e dá às mesmas um novo nível de importância, muito maior que o sugerido em trabalhos como Circunscrição [16] e Lógica Default [19]. Essa importância dada envolve um critério para a montagem das extensões chamado de *exception-first*, de acordo com a construção de extensões para lógica não derrotável proposta em [**?**], onde em vista de que, se as hipóteses de uma regra derivam os limites de outra, então somente o cenário em que a primeira é considerada existe, e um segundo cenário, em que a outra regra seria considerada, é descartado, em contraste com outras propostas, o que ocorre graças ao critério de escolha adotado para a construção de extensões.

Ressaltamos que na LPR, ciclos mútuos (Exemplos 2.1.2, 2.1.2 e 2.1.2) podem gerar múltiplas extensões e ciclos auto-derrotáveis (Exemplo 2.1.2) podem bloquear a existência de qualquer extensão. Uma teoria bem formada não contém ciclos e, portanto, tem sempre apenas uma extensão.

2.1.2 Cenários Plausíveis

Nesse ponto, ressaltamos que os *cenários plausíveis* são compostos por fórmulas em T junto a conjecturas tiradas de coleções maximais de generalizações compatíveis, o que ocorre no caso das teorias bem formadas, caso em que cada generalização no conjunto gerador da extensão (junto às premissas *hard* em T) dá origem a um cenário plausível. Por questões de generalidade e tratamento uniforme, considera-se também as teorias com ciclos defeituosos e as generalizações que causam com que essas teorias sejam assim classificadas. Assim, a definição de cenários plausíveis sugerida funciona nos casos mais gerais em que teorias podem ter mais que uma extensão. Cenários plausíveis são, então, construídos levando em conta somente coleções compatíveis de generalizações que ocorrem em todas as extensões, as *regras disparadas* como definido a seguir.

Definição 2.36. *Dizemos que G' é* disparada em τ *se G' está dentro de E em τ, for cada extensão E em τ. Dizemos ainda que G' é uma regra* maximal disparada em τ *se* G' *é disparada em* τ *e, para cada* G'' *disparada em* τ*, se* $G' \preceq G''$*, então* $G' \approx G''$*.*

Definição 2.37. *Uma generalização é dita* fortemente disparada em τ *se ela pertence a cada* G' *maximal disparada em* τ*.*

Cenários plausíveis são então definidos no caso geral, independente do quão bem a teoria há sido construída.

Definição 2.38. *Um* cenário plausível Σ em τ *é um conjunto de fórmulas tal que* $\Sigma = \mathrm{Th}_{\mathrm{CL}}(T \cup \mathrm{Conj}(G'))$*, onde* G' *é maximal disparada em* τ*.*[14] P vale em um cenário plausível Σ em τ *se* $P \in \Sigma$*. Um* mundo plausível em τ *é qualquer mundo que satisfaça um cenário plausível em* τ*.*

A seguir, são apresentados alguns exemplos para mostrar como os cenários plausíveis são construídos a partir de uma base LPR τ, extraídos do artigo [5].

Suponha que desejamos considerar as informações a seguir:

- Suecos, em geral, são não Católicos.
- Peregrinos de Lourdes, em geral, são Católicos.
- José é um Sueco que participou de uma Peregrinação para Lourdes.

Quais são os cenários plausíveis?

A base LPR $\tau = \langle T, G \rangle$ que representa essas informações é dada a seguir:

$$\begin{aligned} T &= \{\ \mathrm{Swedish(Joseph)}, \mathrm{pilgrim(Joseph)}\ \}; \\ G &= \{\ \big(\mathrm{Swedish}(x) \to \neg\, \mathrm{Catholic}(x)\big) \prec\!\!(\ , \\ &\qquad \big(\mathrm{pilgrim}(x) \to \mathrm{Catholic}(x)\big) \prec\!\!(\ \}. \end{aligned}$$

A única extensão para τ é $\mathrm{Th}_{\mathrm{LPD}}(T \cup \Phi)$, onde

$$\begin{aligned} \Phi = \Big\{ &\Big(\forall x \,\big(\mathrm{Swedish}(x) \to \neg\, \mathrm{Catholic}(x)\big)\Big)?, \\ &\Big(\forall x \,\big(\mathrm{pilgrim}(x) \to \mathrm{Catholic}(x)\big)\Big)? \Big\}. \end{aligned}$$

As duas generalizações em G são incompatíveis e existem dois cenários plausíveis:

$$\begin{aligned} S_1 &= \mathrm{Th}_{\mathrm{CL}}\Big(T \cup \{\ \forall x\, \big(\mathrm{Swedish}(x) \to \neg\, \mathrm{Catholic}(x)\big)\ \}\Big); \\ S_2 &= \mathrm{Th}_{\mathrm{CL}}\Big(T \cup \{\ \forall x\, \big(\mathrm{pilgrim}(x) \to \mathrm{Catholic}(x)\big)\ \}\Big). \end{aligned}$$

No primeiro cenário, é suposto que José não é Católico; no segundo, José é Católico. As duas asserções são plausíveis.

[14] A definição de extensão guarante que um cenário plausível é um conjunto consistente de fórmulas livres de modalidades.

Critério de exceção-primeiro. Suponha que desejamos considerar as seguintes informações:

- Aves em geral Voam, a não ser que sejam Pinguins.
- Pinguins em geral não Voam.
- Tweety e Woody são aves.
- Existe evidência conclusiva de que Tweety é um pinguim.

Quais são os cenários plausíveis?

A base LPR $\tau = \langle T, G \rangle$ que representa essas informações é dada a seguir:

$$\begin{aligned} T &= \{\ \mathrm{bird(Tweety)}, \mathrm{bird(Woody)}\ \}; \\ G &= \{\ \big(\mathrm{bird}(x) \rightarrow \mathrm{flies}(x)\big) \mathrel{-\!\!(} \mathrm{penguin}(x)\,, \\ &\qquad \big(\mathrm{penguin}(x) \rightarrow \neg\, \mathrm{flies}(x)\big) \mathrel{-\!\!(}\,, \\ &\qquad \mathrm{penguin(Tweety)} \mathrel{-\!\!(}\ \}. \end{aligned}$$

A única extensão para τ é $\mathrm{Th}_{\mathrm{LPD}}(T \cup \Gamma)$, onde Γ é

$$\left\{ \Big(\forall x \, \Big(\mathrm{penguin(Tweety)} \wedge \big(\mathrm{penguin}(x) \rightarrow \neg\, \mathrm{flies}(x)\big) \wedge \big(\mathrm{bird(Woody)} \rightarrow \mathrm{flies(Woody)}\big)\Big)\Big) ? \right\}.$$

O único cenário plausível é $\mathrm{Th}_{\mathrm{CL}}(T \cup \Phi)$, onde Φ é

$$\left\{ \forall x \, \Big(\mathrm{penguin(Tweety)} \wedge (\mathrm{penguin}(x) \rightarrow \neg\, \mathrm{flies}(x)\big) \wedge \big(\mathrm{bird(Woody)} \rightarrow \mathrm{flies(Woody)}\big)\Big) \right\}.$$

Nesse cenário, é suposto que Tweety é um pinguim e não voa, enquanto que Woody voa e não é um pinguim.

Repare que um cenário em que Tweety não é um pinguim pinguim e consequentemente voa é contra-intuitivo e não é, portanto, um cenário plausível na LPR, simplesmente porque não deve-se considerar a possibilidade de Tweety não ser pinguim, uma vez que existe evidência de que ele o é. A única razão pela qual se poderia considerar essa possibilidade apesar das evidências, seria no caso de essa evidência vir de uma derivação a partir das primeiras duas generalizações propostas neste exemplo. Nesse caso, a conjectura da primeira regra intervém de forma a bloquear a derivação de sua própria exceção, quando da consideração da segunda. Esse ocorrido é exatamente o que ocorre e prenuncia o critério de exceções-primeiro [18], e explica o motivo porque este não é um cenário aceitável na LPR. As lógicas não monotônicas de Circunscrição [16] e Lógica Default [19] não obedecem ao critério de exceções-primeiro, e na formalização deste exemplo, elas permitiriam aceitar tal cenário. Esse é o motivo pelo qual consideramos que essas lógicas derivam "extensões anômalas" em algumas representações do *frame problem* [8]. No começo da década de 80, esse problema era central em discussões sobre a capacidade e pertinência dos formalismos não monotônicos para representação do conhecimento entre os cientista de Inteligência Artificial, e um estudo detalhado das questões levantadas é feito em [18].

Algumas bases LPR têm mais que uma extensão.

Sejam $T = \emptyset$ e $G = \{\, p \mathrel{-\!\!(} q \,,\, q \mathrel{-\!\!(} p \,\}$.

Essa base tem duas extensões:

$$E_1 = \mathrm{Th}_{\mathrm{LPD}}(\{\, p? \,\})$$

e

$$E_2 = \mathrm{Th}_{\mathrm{LPD}}(\{\, q? \,\}).$$

Essa base é defeituosa no sentido de que as duas generalizações se excluem mutuamente, uma levando à derivação da restrição da outra, o que vem a caracterizar um ciclo mútuo. No ponto de vista dos autores de [5], esses ciclos não fazem sentido, e representa uma falha na representação do problema. De acordo com a abordagem proposta neste trabalho, nem $\mathrm{Th}_{\mathrm{CL}}(\{\, p \,\})$ nem $\mathrm{Th}_{\mathrm{CL}}(\{\, q \,\})$ é suficiente para formar um cenário plausível, pois cenários plausíveis são feitos a partir de presentes em todas as extensões.

Algumas bases LPR não têm nenhuma extensão.

Se $T = \emptyset$ e $G = \{\, p \mathrel{-\!\!(} q \,,\, q \mathrel{-\!\!(} r \,,\, r \mathrel{-\!\!(} p \,\}$, então τ não apresenta nenhuma extensão. A teoria resultante apresenta um ciclo auto-derrotável e se trata de uma teoria defeituosa. As três generalizações formam um ciclo bloqueando o uso de qualquer uma delas. Supõe-se que essas generalizações não contém informações relevantes, ou que elas simplesmente são descritas a partir de erros de concepção do problema por parte de quem propõe a base.

Algumas bases LPR têm uma extensão cujo conjunto de regras geradoras é reduzido a $\{\emptyset\}$.

Se $T = \emptyset$ e $G = \{\, p \mathrel{-\!\!(} p \,\}$, então $E = \mathrm{Th}_{\mathrm{LPD}}(\emptyset)$ é a única extensão de τ, e $\mathrm{Th}_{\mathrm{CL}}(\emptyset)$ é o seu cenário plausível correspondente. Repare que $\overline{\Psi}_\tau(E) = \{\emptyset\}$. Novamente, consideramos essa base como defeituosa, pois ela apresenta uma generalização levando à derivação de suas próprias restrições, o que caracteriza um ciclo auto-derrotável. A generalização "$p \mathrel{-\!\!(} p$" é sem sentido e sem uso prático.

Se $T = \emptyset$ e $G = \{\, p \mathrel{-\!\!(} q,\, p \to q \mathrel{-\!\!(} \,\}$, então existe uma única extensão E tal que $\overline{\Psi}_\tau(E) = \big\{ \{\, p \mathrel{-\!\!(} q \,\}, \{\, p \to q \mathrel{-\!\!(} \,\} \big\}$. Mais um vez, essa base é defeituosa, pois ela apresenta uma generalização levando à derivação de suas próprias restrições.

A *teoria gerada por uma* base LPR τ é definida a seguir. Repare que, se τ tem ao menos uma extensão, as fórmulas em T são necessárias, as sentenças consistentes com T são possíveis, as fórmulas plausíveis valem em algum cenário plausível e as fórmulas estritamente plausíveis valem em todos os cenários plausíveis.

Definição 2.39. A teoria gerada por uma *base* LPR $\tau = \langle T, G\rangle$, *denotada por* $\Pi(\tau)$, *é a menor coleção de fórmulas de* L *que satisfaz as seguintes condições:*

- $T \subseteq \Pi(\tau)$;[15]
- *Se* $\Pi(\tau) \mid\!\frac{}{\mathrm{LPD}} P$, *então* $P \in \Pi(\tau)$;[16]
- *Se* P *é uma sentença livre de modalidades e* $T \cup \{P\}$ *não é* LPD*-trivial, então* $\Diamond P \in \Pi(\tau)$;[17]
- *Se* G' *é finita e disparada em* τ, *então* $(\bigwedge \mathrm{Conj}(G'))? \in \Pi(\tau)$;[18]
- *Se* $P \prec\!\!\!(Q$ *é fortemente disparada em* τ, *então* $P! \in \Pi(\tau)$.[19]

Os elementos de $\Pi(\tau)$ *são também chamados* teoremas de τ.

Teorema 2.20. $\Pi(\tau) = \mathrm{Th}_{\mathrm{LPD}}(T \cup T_1 \cup T_2 \cup T_3)$, *onde:*

- $T_1 = \{\, \Diamond P \mid P$ *é uma sentença livre de modalidades e* $T \cup \{P\}$ *não é* LPD*-trivial* $\}$;
- $T_2 = \{\, (\bigwedge \mathrm{Conj}\,(G'))? \mid G'$ *é finita e disparada em* $\tau \,\}$;
- $T_3 = \{\, P! \mid$ *existe* Q *tal que* $P \prec\!\!\!(Q$ *é fortemente disparada em* $\tau \,\}$.

Definição 2.40. $\tau \mid\!\frac{}{\mathrm{LPR}} P \rightleftharpoons P \in \Pi(\tau)$.

As quatro modalidades mantém em LPR um relacionamento análogo àquele expresso para a LPD no teorema 2.11.

Agora podemos definir uma estrutura-LPD que nos permite raciocinar com os cenários possíveis e plausíveis induzidos pelas premissas hard e soft de uma dada teoria LPR.

Definição 2.41. *Uma estrutura*-LPD $H = \langle \Delta, W, W'\rangle$ para L *é dita* satisfazer *a uma base* LPR τ *se as seguintes condições são alcançadas:*

[15]Ou seja, as premissas *hard* são teoremas de τ.

[16]Ou seja, o conjunto de teoremas de τ é fechado para a dedução em LPD.

[17]Ou seja, $\Diamond P$ é um teorema de τ, para todas as sentenças livres de modalidades P consistentes com T na LPD.

[18]Existe uma regra maximal disparada G'' em τ de forma que $G' \preceq G''$ e $\bigwedge \mathrm{Conj}(G')$ valem no cenário plausível $\mathrm{Th}_{\mathrm{CL}}(T \cup \mathrm{Conj}(G''))$.

[19]Ou seja, P pertence a todos os cenários plausíveis.

- *H satisfaz T;*[20]
- *para cada sentença livre de modalidades P, se $T \cup \{P\}$ é não* LPD-*trivial, então H satisfaz $\Diamond P$;*[21]
- *cada cenário plausível em τ é satisfeito por algum mundo plausível $w' \in W'$;*
- *Todo mundo plausível $w' \in W'$ satisfaz a algum cenário plausível em τ.*

Definição 2.42. *Dizemos que uma fórmula P é uma* LPR-consequência semântica de *uma base* LPR *τ se cada estrutura*-LPD *H que satisfaz τ também satisfaz P. Quando isso acontece, utilizamos a notação $\tau \models_{\overline{\mathrm{LPR}}} P$ para dizê-lo.*

Provabilidade e consequência semântica têm a mesma extensão (em termos de teoria dos conjuntos) para a LPR.

Teorema 2.21 (Corretude e Completude da LPR)**.**
- *$\tau \vdash_{\overline{\mathrm{LPR}}} P$ sse $\tau \models_{\overline{\mathrm{LPR}}} P$.*

s Apenas repare que uma estrutura-LPD satisfaz τ se, e somente se, ela satisfaz $T \cup T_1 \cup T_2 \cup T_3$, como definido no teorema 2.20, e que a LPD é correta e completa, de acordo com o teorema 2.21.

Agora estamos prontos para construir uma teoria e definir uma base LPR adequada para trabalharmos esses conceitos com mais detalhe na prática. A seguir, apresentaremos o propósito geral deste trabalho e seus resultados, utilizando os conceitos aqui apresentados para gerarmos uma extensão e conduzirmos o problema à conclusão de um cenário plausível e, consequentemente, uma solução viável para este problema, ou algumas alternativas, caso necessário.

[20] Ou seja, as fórmulas em T são satisfeitas por todos os mundos possíveis $w \in W$.
[21] Ou seja, P é satisfeita por algum mundo possível $w \in W$.

Capítulo 3

Aplicação da LPR

Após introduzir detalhadamente as lógicas relacionadas LPR e LPD, nos resta mostrar uma aplicação da mesma em ambiente lógico modelado e processado pelas mesmas. Em vista da recente publicação do trabalho [5] que introduz tais lógicas, e tendo em vista dar continuidade ao trabalho elaborado neste, buscamos introduzir uma aplicação exemplo de uso dos mecanismos apresentados para mostrar como se dão os resultados do cálculo de extensões e como estas são geradas em algumas instâncias de problemas. Dessa forma, objetivamos como pontos principais deste um estudo aprofundado de lógica não monotônica através da LPR e a aplicação da mesma para o cálculo de extensões em um problema que nos permite ponderar e fazer observações sobre o problema a partir de informações obtidas através de inferências ampliativas. A modelagem apresentada a seguir apresenta maior robustez e complexidade comparada aos exemplos simplificados de [5]. Além disso, nosso trabalho mostra algumas variações do problema para produzir diferentes resultados, ilustrando de forma diversificada o uso da LPR e propõe, para o problema geral, uma função de utilidade bastante simples, fundamentada na hierarquia de predicados propostos de acordo com a proximidade de uma resposta decisiva, compondo então um sistema de suporte à decisão baseado na LPR.

Como sugerido em exemplo do raciocínio indutivo, utilizamos uma formalização em regras do que seria uma busca investigativa. Os contextos sugeridos para a investigação são instâncias de problemas de um jogo, e as regras que montam as possibilidades e condições de acusação e das ocorrências dentro do mesmo são detalhadas pelas regras do jogo, extraídas a partir da prática e do auxílio de tabelas de acompanhamento. Uma vez que algumas instâncias do problema apresentadas pelo jogo em níveis de dificuldade mais avançados tende a ser normalmente inconclusiva, nos resta proceder com suposições, e a partir de então necessitamos de um formalismo lógico dotado da característica ampliativa para retratar tais inferências. O interesse na LPR procede da continuidade de um trabalho em iniciação científica envolvendo o estudo de sua predecessora, IDL, bem como na funcionalidade dos conceitos de generalizações e extensões e seu poder, ainda muito pouco explorados, uma vez que o presente é um dos primeiros trabalhos a abordar tal formalismo, o que é ao mesmo tempo uma facilidade e um desafio. Facilidade por termos ainda muito espaço para abordar e ponderar sobre os mecanismos utilizados, e um desafio por não haverem outros trabalhos que sirvam como base e suporte a essas observações contidas em todo o escopo do presente texto.

Para abordar um contexto de "adivinhação" em que sabe-se somente existir uma resposta, mas em que os dados disponíveis são inconclusivos e, portanto, não apontam para nenhuma das possíveis respostas, cabe-nos gerir suposições e evidências de forma interessante (para a obtenção de uma resposta), consistente e formal. A parte que compõe o quão interessante é a nossa abordagem nesse ponto de vista depende diretamente dos predicados e relações criadas entre os mesmos. A consistência e o formal são importados da LPR, escolhida como mecanismo devido ao seu poder de inferência indutiva e por trabalhar principalmente com conclusões diretas, dadas suposições bem definidas. Esse aspecto é retratado na estratégia da LPR de derivar extensões maximais, priorizando as exceções em cada instante, para então utilizarmos somente regras de inferência dedutivas sobre o contexto suposto [1]. Dessa forma, como as conclusões e consequente decisão a serem tomadas seguem de um processo inicialmente indutivo e depois dedutivo quando processados pelas lógicas aqui referenciadas, adotamos as mesmas como ferramentas principais na produção deste sistema de apoio à decisão.

Outra vantagem obtida com o uso da LPR no tratamento deste problema é que o cálculo de extensões leva em conta todas as possibilidades de extensões para calcular aquela que é mais abrangente. Dessa forma, o domínio do problema é totalmente coberto e trabalhamos com a melhor extensão para cada instância do mesmo, visto que alguns dados das instâncias influenciam na prova de exceções de algumas regras. A própria proposta algorítmica recursiva para a criação de extensões visando ponto-fixo é realizada de forma que o problema é computacionalmente equivalente a 3-SAT, o que se deve à

[1] Supomos a extensão maximal, pois essa seria a menos propensa a retrocessos na teoria, e dessa forma seria a mais geral. Tal extensão é então composta por várias suposições e constitui de um contexto unificado, suportado pelas generalizações e que não contém absurdos implícitos derivados das exceções de diferentes regras. Uma vez que seus componentes identificadores são extraídos de regras ampliativas, o conjunto como um todo é ampliativo e, portanto, uma grande suposição.

natureza combinatória do problema sobre o número de regras de generalização envolvidas [2].

Seguiremos este capítulo com a apresentação das regras e especificações do jogo que deu origem à aplicação, em função de levar à melhor compreensão da aplicação aqui apresentada. Na sequência, especificaremos a modelagem lógica sugerida para a representação matemática das regras envolvidas e comentaremos sobre alguns pontos da mesma. Algumas modificações são feitas nesta fase para tornar o problema ligeiramente mais complexo e, dessa forma, mais interessante para nosso trabalho. A função de utilidade é comentada logo após a explicação dos predicados criados para fundamentar diferentes níveis de "culpa", as quais são bem presentes nos casos não conclusivos, e reforça a intenção por trás da hierarquia de força expressiva dos predicados sugeridos. Para acompanhar o processo de cálculo de extensões, escrevemos um programa em prolog que realizar algumas deduções na base de dados e a alteramos conforme necessário, incluindo algumas conclusões, ambas dedutivas e ampliativas, e realizando novas interações afim de encontrar alguma resposta dentro da escala sugerida que possamos utilizar, aplicando a função de utilidade definida para a decisão final do sistema. Isso conclui a apresentação dos passos seguidos para a produção deste auxiliar de decisão, faltando somente comentar os resultados obtidos e realizar algumas observações mais sobre a performance e completude do processo.

3.1 O problema abordado

Os jogos de tauleiro de detetive são um clássico da indústria que os fez popular através do título *Clue*. No Brasil, o jogo veio com o nome *Detetive*, e foi bastante vendido. *Sleuth* é um jogo no mesmo gênero, incluindo os cenários equivalentes a um tabuleiro, mas sem a competição entre vários jogadores, nem as obrigatoriedades inclusas pela competição, como um dado ou roleta para determinar possibilidades de ações e revezamento de turnos. Sendo feito para apenas um jogador em seu modo de jogo principal [3], as instâncias de casos propostos são gerados por sorteio através de funções randômicas de computador, limitadas apenas por algumas regras que regem a garantia de

[2]Em verdade, a complexidade do cálculo é duplamente exponencial, uma vez que são verificados todas regras como subconjuntos do conjunto de generalizações, e em seguida, levadas em conta todas as sub-regras de cada regra, denotando o conjunto das partes das partes das generalizações 2.25 e 2.29.

[3]*Sleuth Noir* propõe uma modalidade cooperativa para assinantes através da criação de agências de detetives para resolver casos mais complexos. Nossa proposta é com relação ao raciocínio desenvolvido visando a solução de um enigma do jogo solo.

existência de uma solução, e o nível de dificuldade dos casos. Muitos são os dados envolvidos na criação, bem como as possibilidades de casos sugeridos pelo jogo. Nossa modelagem, a ser proposta em seguida à introdução do jogo, visa recriar o raciocínio envolvido após a coleta de informações, uma vez que os dados obtidos variam de acordo com as decisões do jogador, e o framework lógico proposto para a solução, a LPR, não tem previsão de interatividade. Essa interatividade poderia ser implementada, sugerindo uma aplicação de jogo eletrônico seguindo o gênero *dungeon*, muito popular nos tempos de sistemas operacionais desprovidos de interface gráfica. Várias suposições envolvendo ações dos jogadores são omitidas na modelagem final, uma vez que é de nosso interesse inicial mostrar somente a capacidade da LPR que gerar conclusões a partir de um conjunto de regras LPR.

Sleuth é um jogo online encontrado em 'http://noir.playsleuth.com/login.spy', com inscrição gratuita que dá acesso ao jogo solo. Esse registro para o jogo gratuito limita o número de casos resolvidos por dia a quatro, e somente permite jogar aos casos gerados por computador com roteiros padronizados. O registro pago permite jogar muitos mais casos ao dia, bem como se juntar a uma agência de detetives para resolver casos em conjunto com outros jogadores, e resolver casos com roteiro personalizado, escritos ou aprovados pelos autores do jogo. Casos personalizados não são criados através de regras matemáticas como os gerados por computador, e exigem intuição, atenção e inteligência dos jogadores, visando colocá-los em um ambiente similar ao de um detetive das histórias consagradas de autores consagrados do gênero de litaratura como Agatha Christie e Conan Doyle, histórias nas quais se inspira o ambiente do jogo. Sendo um RPG (Role Playing Game), a imersão dos jogadores é um fator muito importante, a qual é alcançada nos jogos padronizados através de detalhamento dos roteiros, imagens de NPC's (non player characters, ou personagens não-jogáveis, que não são dos jogadores e servem para compor cenários), interatividade com o mapa e ambientes variados na solução dos casos. Em ambos os modos, o jogador se coloca no lugar de um detetive, e passa a administrar pistas e suspeitos em função de encontrar uma solução para o caso. Em jogos solo de casos gerados (os que realmente nos interessam, devido à maneira matemática como são montados, sugerindo regras conhecidas) temos sempre um assassinato, e uma pessoa relacionada à vitima procura pelos serviços do personagem em busca de uma solução (Veja a figura 3.1). O contratante indica o nome de algumas pessoas de quem suspeita, e é considerado também um suspeito, pois às vezes o próprio assassino ordena a investigação.

A partir da descrição do caso fornecida pelo contratante, temos uma lista

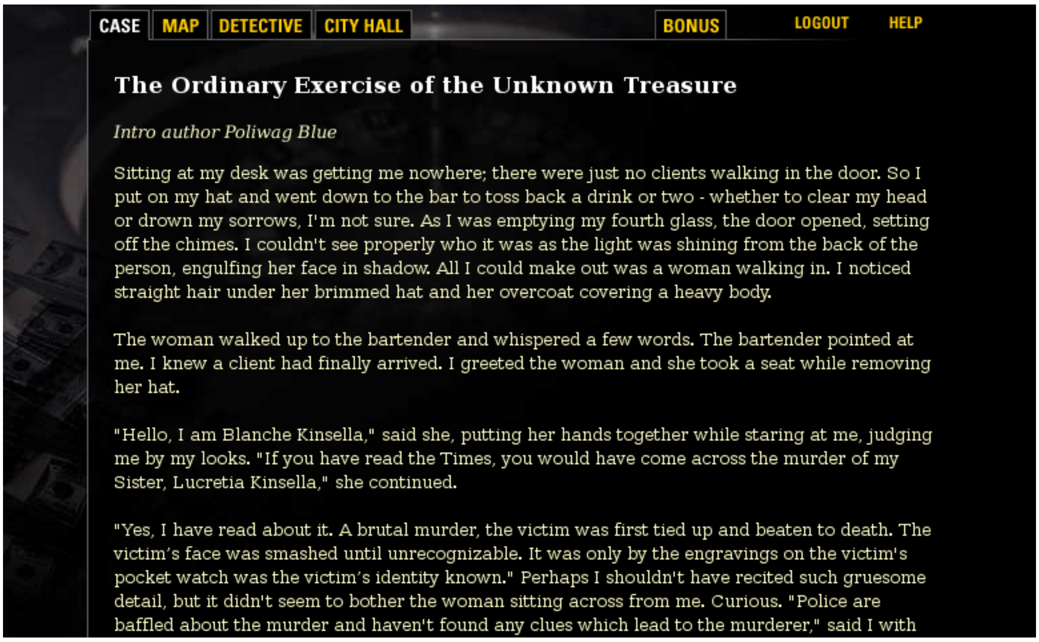

Figura 3.1: Contexto do caso.

inicial de pessoas para abordar em entrevista. A lista é composta por nossos primeiros suspeitos: A pessoa que solicita a investigação, e aquelas apontadas por ela como possíveis assassinos (Veja a figura 3.2).

Em entrevista, é possível descobrir outros suspeitos, apontados por estes, sempre de acordo com uma possível causa sugerida em conjunto. Essas causas variam entre dívidas, vingança, desgosto pela vítima, herança prometida, entre outros. Para garantirmos a solução e termos certeza de que podemos encontrar o assassino, uma tarefa essencial é encontrarmos todos os possíveis suspeitos. A descrição inicial do crime também sugere uma cena, a qual pode ser visitada em busca de evidências físicas que indicam a presença de algumas pessoas entre os suspeitos que estiveram no local recentemente, o que reforça as possibilidades de um destes ser o culpado. Em verdade, enquanto cada evidência somente está associada a uma pessoa, uma dessas foi, com certeza, deixada na cena pelo assassino (Veja a figura 3.3).

As evidências são mais facilmente encontradas e analisadas corretamente para extrair uma característica do suspeito relacionado à mesma se o detetive tiver as habilidades (skills), adequadas (Veja a figura 3.4).

Como comumente imposto por jogos desse gênero, os ambientes de investi-

Figura 3.2: Primeiras notas sobre o caso.

Figura 3.3: Cena do crime.

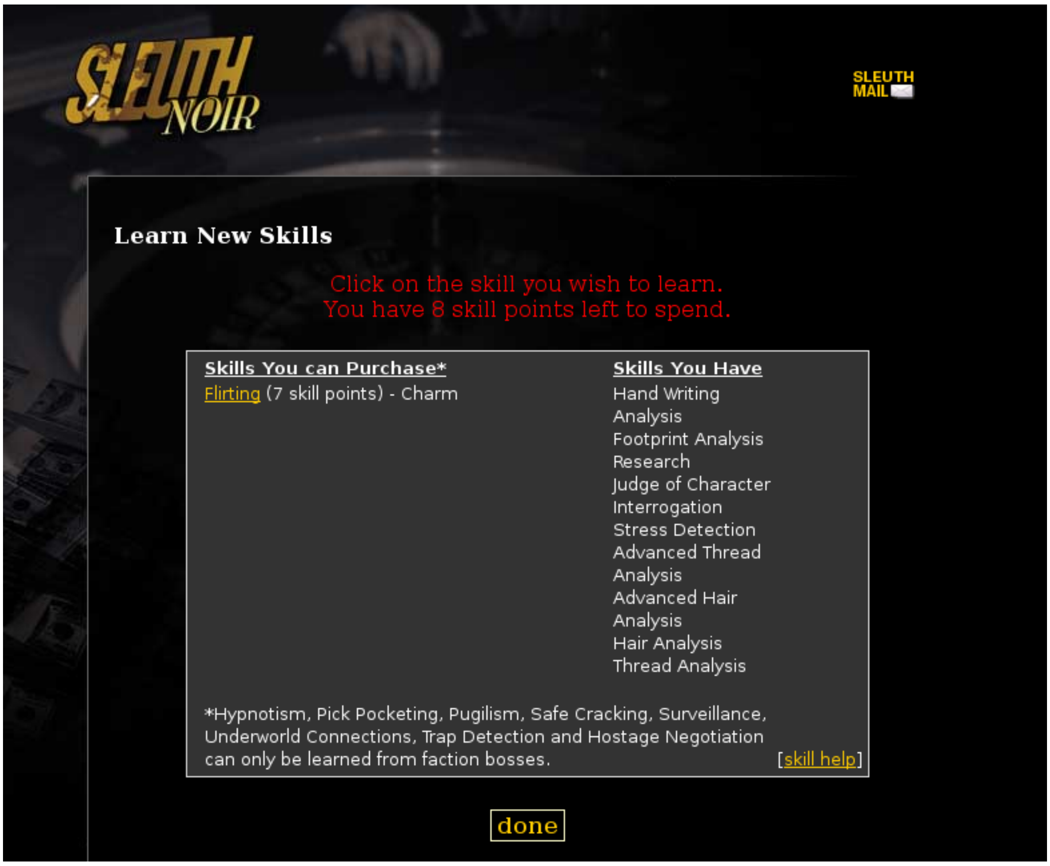

Figura 3.4: Habilidades do detetive.

Figura 3.5: Mapa da cidade.

gação são limitados. No caso de *Sleuth*, supomos uma cidade relativamente pequena, em uma época passada, onde todos se conhecem (Veja a figura 3.5).

É nesse ponto que são introduzidos os NPCs, especialmente aqueles que ajudam ao detetive com evidências físicas. A cidade conta com vários ambientes, cada qual com algum NPR responsável pelo mesmo, e que serão utilizados como álibis pelos suspeitos (algumas vezes mentirosos). Nesses locais, podemos conversar com os personagens em função de conferir álibis, perguntar por rumores (o que pode nos dar o nome de outros suspeitos ou indicação de que alguém tem mais informações), ou, em alguns destes, verificar se as evidências pertencem a uma ou outra pessoa (Veja a figura 3.6). Os NPC's que verificam evidências são:

- O *Barbeiro*, que corta os cabelos de todos da cidade e pode comparar fios encontrados na cena do crime com os dos suspeitos;
- O *Alfaiate*, que lida com as roupas de todos e reconhece fibras de tecido utilizadas nas roupas dos suspeitos;
- O *Sapateiro*, que faz os calçados de todos e consegue reconhecer as pegadas dos suspeitos;
- O *Banqueiro*, que administra o dinheiro das pessoas da cidade e tem assinatura registrada de todos os suspeitos, podendo, então, comparar a escrita à mão dos suspeitos com notas encontradas na cena do crime.

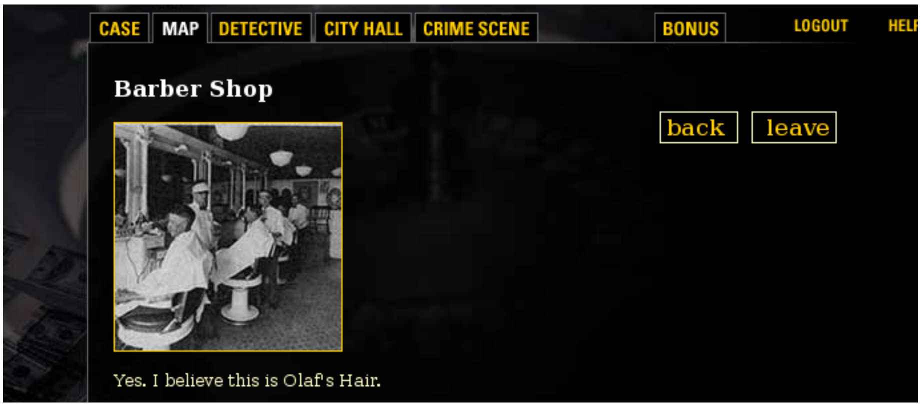

Figura 3.6: Combinação prova-suspeito.

.
As evidências encontradas indicam algumas características possíveis do assassino. São elas:

- Fios de cabelo indicam a presença recente de um suspeito de cabelos *lisos* ou *crespos* na cena do crime.
- Fibras de roupa indicam a presença recente de um suspeito do sexo *masculino* ou *feminino* na cena do crime.
- Pegadas indicam a presença recente de um suspeito *leve* ou *pesado* na cena do crime.
- Notas escritas indicam a presença recente de um suspeito *canhoto* ou *destro* na cena do crime.

As características dos suspeitos são averiguadas pelo detetive no momento da primeira entrevista. Nesse ponto, o personagem do jogador descobre se o suspeito é homem ou mulher, canhoto ou destro, etc... As características são agrupadas duas a duas em condições mutuamente exclusivas, e sabemos que as evidências só podem pertencer a algum suspeito que tenha características compatíveis com as sugeridas pela peça de evidência. As relações ocorrem da seguinte forma:

- Fios de cabelo crespos somente podem pertencer a suspeitos de cabelos crespos;
- Fios de cabelo lisos somente podem pertencer a suspeitos de cabelos lisos;
- Fibras que indicam roupas masculinas somente podem pertencer a sus-

peitos homens;
- Fibras que indicam roupas femininas somente podem pertencer a suspeitos mulheres;
- Pegadas de pessoas pesadas somente podem ter sido produzidas por suspeitos de constituição pesada;
- Pegadas de pessoas leves somente podem ter sido produzidas por suspeitos de constituição leve;
- Notas escritas por pessoas canhotas só podem ter sido produzidas por suspeitos canhotos;
- Notas escritas por pessoas destras só podem ter sido produzidas por suspeitos destras;

Tendo em mente que as características de cada suspeito são facilmente obtidas através de contato visual na primeira entrevista, já podemos inclusive inocentar algum suspeito, caso nenhuma das evidências possa ter sido produzida por ele. Nessas entrevistas aos suspeitos, e nos contatos com os NPCs, cada personagem envolvido tem disposição para responder somente um número limite de perguntar, sorteado na construção do caso, até um máximo de três perguntas. Quanto mais alta a dificuldade do caso, menos perguntas os suspeitos e personagens vão querer responder, o que indica menos respostas e menos informações. As exceções disso são personagens para quem fazemos favores e que se tornam nossos contatos. É permitido ao detetive ter um contato que faça análise de evidências, e um contato que possa prover rumores. Os primeiros são os personagens citados anteriormente, e os demais são responsáveis por ambientes da cidade comuns, como o açougue, o restaurante, o bar, a igreja, a escola de música e o cigano. Todos os personagens da cidade, de ambos os tipos, podem ser citados como companhia em um álibi dos suspeitos, e portanto, responder à confirmação de um álibi quando indagados (se dispostos a responder) (Veja a figura 3.7). Os contatos respondem a qualquer número de perguntas que desejarmos fazer.

.

As investigações em *Sleuth* se dão de forma que o detetive pode buscar novas informações com os suspeitos e NPC's, bem como confirmá-las ou negá-las em função de descobrir o culpado de crime. Dependendo das habilidades do detetive, é possível realizar pesquisa por ocorrências envolvendo os suspeitos no momento do crime (o que provaria a inocência dos mesmos, uma vez que não seria possível estar simultaneamente na cena do crime e em um incidente documentado em outro lugar), conseguir informações extras além da disposição dos personagens entrevistados, seja por conversa e simpatia ou intimidação e ameaças, possibilidades que cobrem os principais estereótipos de personagens de histórias de suspense do gênero, conhecer testemunhos de

Figura 3.7: Confirmação de um álibi.

alguns personagens contra outros, ou ainda descobrir mais informações através de análise das provas. As chances de convencer cada suspeito ou NPC é igualmente sorteada na formulação do caso, e influenciam diretamente para tornar os casos mais fáceis ou difíceis (com chances maiores ou menores de persuadir os personagens). As ações são bem definidas e, consequentemente, todas as informações relevantes podem ser enumeradas:

- Os nomes e características de todos os suspeitos. Os nomes servem de referência aos mesmos, enquanto que as características vão permitir relacionarmos as evidências encontradas aos possíveis suspeitos que a produziram.
- As evidências físicas e características das mesmas. Similar ao que ocorre com os suspeitos, utilizamos as características das evidências para buscar os suspeitos que estiveram na cena do crime.
- As razões que cada personagem teria para cometer o crime. Estas informações são requisito para podermos fazer a acusação. Isso quer dizer que se não conhecermos os motivos do assassino, a acusação não pode ser realizada por não ter credibilidade. As regras que montam o conjunto de dados do jogo permite descobrirmos o assassino mesmo sem conhecer seu motivo.
- Os casos em que soubermos que uma evidência pertence a um suspeito em específico, especialmente se soubermos qual das evidências foi deixada pelo culpado.
- Os álibis dos suspeitos. Estes servem para confirmar a inocência dos mesmos. O assassino não tem álibi ou fornece um falso.
- Testemunhos sobre ações estranhas dos suspeitos por parte de outros suspeitos (Como se ter sido visto ao livrar de uma arma, ter sido visto

Figura 3.8: Pesquisa.

sujo de sangue, etc).

- A ocorrência de algum incidente no momento do crime que possa inocentar um dos suspeitos (Veja a figura 3.8). Os suspeitos envolvidos em escândalos ou acidentes normalmente fornecem alibis falsos para não citar o ocorrido.
- A morte de um suspeito durante a investigação. Essa informação indica inocência do suspeito morto, crime cometido pelo mesmo assassino, possivelmente para eliminar uma testemunha (Veja a figura 3.9).

.

.

No jogo *Sleuth*, temos a possibilidade de escolher a dificuldade de cada caso que desejemos resolver. Existem 10 dificuldades (Veja a figura 3.10) variando de "principiante" a "quase impossível", que representam conjuntos de parâmetros diferentes para a criação aleatória dos casos, de forma a aumentar cada vez mais a quantidade de suspeitos e evidências a serem combinadas, e reduzindo as probabilidades de conseguirmos ajuda e o número de perguntas respondidas por cada personagem. Ascender aos níveis mais altos e ganhar acesso aos casos mais difíceis é demorado, e por isso utilizaremos o conceito do nível "incrivelmente difícil", onde já encontramos o máximo de evidências (quatro) e temos um universo de nove suspeitos (o limite são onze), e uma grande chance de iniciarmos o caso e, ao fim de todas as ações possíveis, entrevistas, pesquisas e consultas aos NPC's, terminarmos com informações insuficientes para uma conclusão definitiva do caso, ou seja, terminamos de

Figura 3.9: Suspeito morto durante a investigação.

forma que ainda temos dois ou mais suspeitos que podem ter cometido o crime, e sabemos que só um deles é culpado. O jogo é definido de forma que os motivos dos crimes, independente de quais sejam, são todos argumentos de mesma força, bem como os álibis verdeiros ou falsos, ou qualquer outra informação que reflita em inocência ou culpa dos suspeitos, mais uma vez, sugerindo predicados muito bem definidos. A inocência dos suspeitos pode ser constatada através da verificação de alguma das seguintes ocasiões:

.

- Um álibi verdadeiro prova a inocência do suspeito;
- A morte de um suspeito prova a inocência do mesmo (pois significa que o culpado era outro entre os suspeitos);
- Se for comprovado que nenhuma das evidências encontradas na cena do crime foi deixada ou produzida por um dos suspeitos, está provada a inocência do mesmo;
- Uma pesquisa que encontre histórico de algum incidente envolvendo o suspeito também comprova sua inocência;

O assassino pode ser detectado através de uma das seguintes ocasiões, se verificadas:

- Se conhecemos todos os suspeitos e constatamos que todos menos um destes é inocente, então o que sobra é o assassino (Pois é garantido que o assassino está entre os suspeitos);

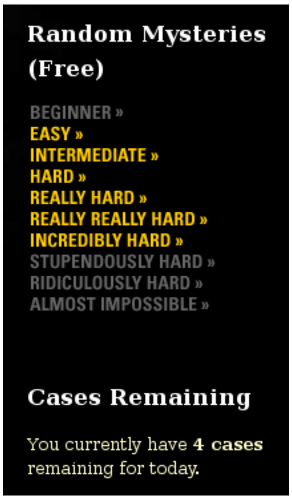

Figura 3.10: Níveis de dificuldade.

- O assassino tem um álibi falso (ou não apresenta nenhum álibi) e sabemos que alguma das evidências na cena do crime foi produzida por este;
- O assassino tem um álibi falso (ou não apresenta nenhum álibi) e algum outro suspeito testemunha contra ele, indicando novas evidências da ocorrência do crime por sua parte.
- Sabemos que alguma das evidências na cena do crime foi produzida por este e dois outros suspeitos testemunham contra ele, indicando novas evidências da ocorrência do crime por sua parte.

Em verdade, as três condições (evidência, duplo testemunho, álibi falso) se verificam para o assassino, mas nem sempre é possível determinar as três condições e nos bastam apenas duas para incriminá-lo com certeza, pois inocentes nunca cumprem o requisito de duas dessas condições simultaneamente. Termos somente um testemunho contra um suspeito evidência suficiente de culpa quando conhecemos falsidade do álibi, e forte evidência de culpa quando sabemos ter uma evidência física própria do mesmo. Podemos dizer que estamos muito próximos de uma conclusão neste segundo caso, ou seja, se não tivermos certeza, um testemunho somente, combinado com um dos outros fatores que indicam culpa tendem a ser nossa melhor aposta em uma acusação. Os casos são encerrados quando o detetive faz uma acusação a um dos suspeitos e mostra que o mesmo é o culpado, ou faz uma acusação errônea e é a afastado do caso. De uma forma ou de outra, somente uma acusação é permitida (Veja as figuras 3.11 e 3.12).

.

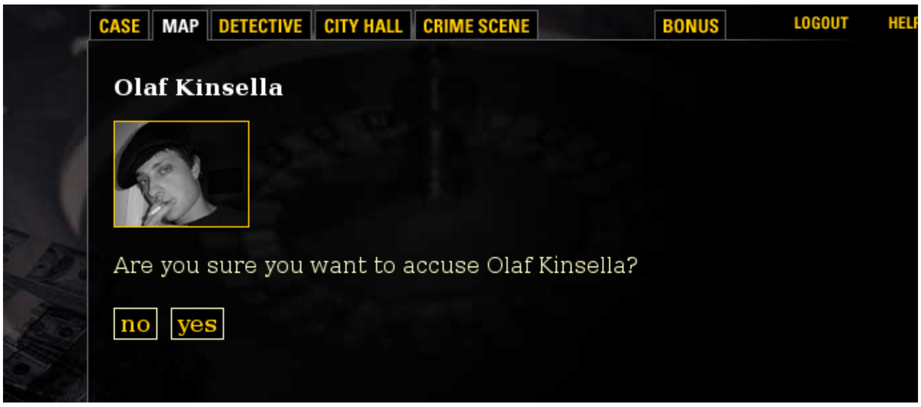

Figura 3.11: Acusação.

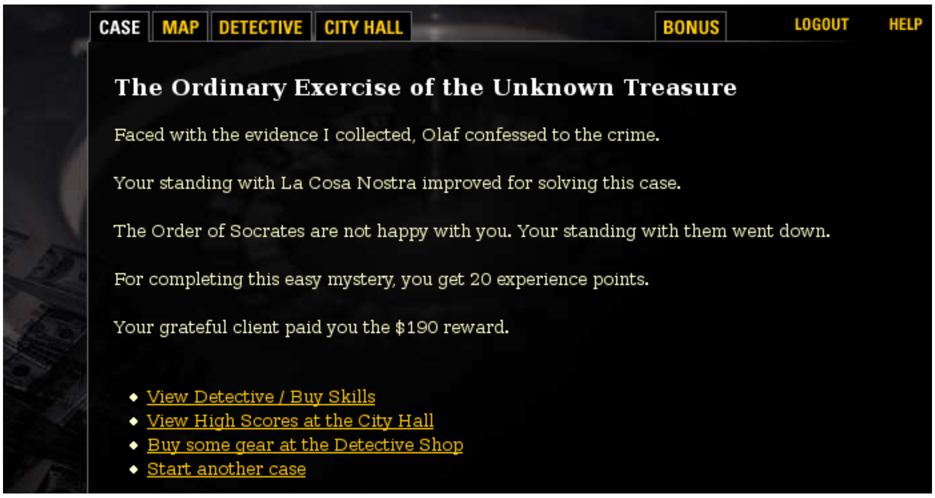

Figura 3.12: Encerramento do caso.

.
Invés de realizar uma acusação incerta, é possível simplesmente desistir do caso, o que pode vir a ser boa idéia caso o detetive esteja procurando criar renome junto a alguma facção atuante na cidade. No modo gratuito de jogo de Sleuth temos somente uma cidade, com operações de somente três facções. Vez por outra, durante os casos, receberemos notícias ou descobriremos fatores do caso que influenciam junto a essas facções. Normalmente, resolver o caso seria bom no ponto de vista de uma delas e ruim no ponto de vista de outra, e se a resolução for ruim para a facção na qual o detetive pleiteia contatos, pode ser melhor apenas ignorar o caso e se afastar dele, cancelando as investigações. Cada facção fornece algumas vantagens únicas, incluindo algumas habilidades que somente podem ser adquiridas uma vez alcançado um

alto status dentro destes grupos, quando o detetive se torna suficientemente confiável para aprender mais sobre os segredos da facção. Os grupos citados são nomes conhecidos, passando por grupos mafiosos criados pela literatura ou mesmo algum que supõe-se realmente existir, tais como 'La Cosa Nostra', 'A mão verde', 'Shangri La' e a 'Ordem de Sócrates'.
Além de apresentar várias possíveis situações e a possibilidade de análise de conjuntos de informações crescente e suscetível a regras bem definidas, observáveis da própria prática com o jogo, temos como instrumento de medição e verificação da aplicabilidade das regras o tutorial do jogo e uma ferramenta fornecida por fãs. *Doctor Watson* é um site à parte, encontrado em "http://www.angus.demon.co.uk/sleuth/watson.htm" e citado em *Sleuth* entre outros suplementos produzidos por fãs, que fornece uma ferramenta tabular para anotar e administrar os dados conseguidos através dos várias passos de investigação (Veja a figura 3.13). Além de permitirnos organizar os dados em uma planilha propriamente montada, com representação da maior parte das provas e características pertinentes à cena do crime e aos suspeitos, *Doctor Watson* nos fornece ajuda processando automaticamente algumas das regras e apontando o culpado em caso evidência definitiva. Essa ferramenta tanto nos fornece um novo ponto de vista do jogo através da percepção que temos de algumas regras implícita, como também permite avaliar o quadro composto por evidências, checagem de álibis e etc de forma facilitada. Como proposto para auxiliar aos jogadores na resolução de casos, a ferramente é de grande ajuda, especialmente em níveis mais difíceis, em que as informações organizadas, já que escassas, são essenciais. Essa facilidade também nos servirá de ferramenta para a averiguação dos resultados alcançados com nossa proposta.

.

Escolhemos *Sleuth* como inspiração para uma aplicação de nossa lógica por apresentar uma série de características pertinentes que permitiriam ilustrar as várias facilidades e a capacidade expressiva da LPR em termos de generalizações e representação do raciocínio. Além da enorme variação de instâncias de problemas possíveis para aplicarmos o modelo de resolução sugerido a seguir, temos níveis de dificuldade em que a resolução dos casos em definitivo é dificultada e depende de sorte nas escolhas sobre a utilização dos recursos de investigação (perguntas a personagens, consultas a NPC's, etc...), o que nos trás a instâncias de problema que nos obrigam a fazer suposições sobre a parte complementar desconhecida da base de dados em função de chegarmos uma conclusão. Essa característica de conjuntos de informações insuficientes sugere um tratamento de prova indutiva. A necessidade de regras formais bem definidas para a possibilidade de gerar casos aleatoriamente complementa com uma facilidade para a modelagem e tratamento por lógica,

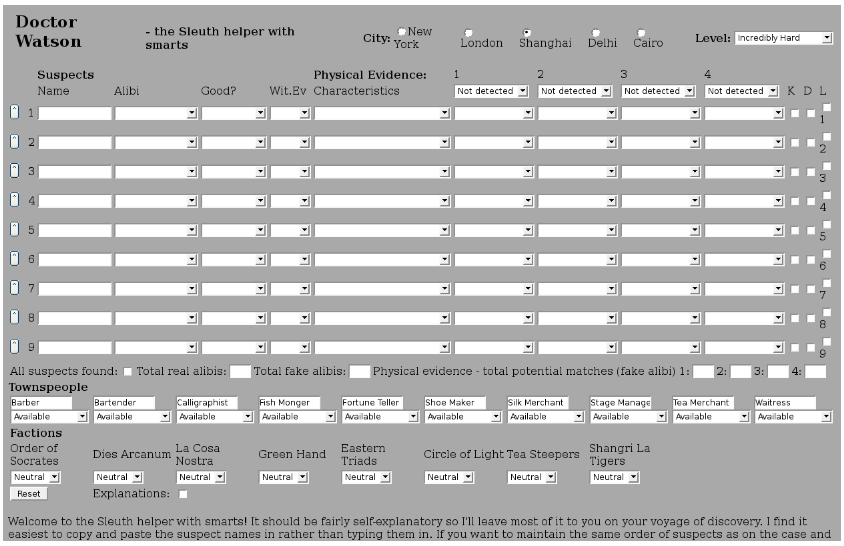

Figura 3.13: Dr. Watson

e sugere grande parte do modelo. Defendemos que *Sleuth* é uma excelente aplicação para trabalhar a LPR em função dessas características e convidamos o leitor a prosseguir e conhecer nossa proposta para a formalização das regras de Sleuth em lógica de primeira ordem, utilizando a notação estendida para generalizações proposta na LPR.

Capítulo 4

Modelagem e cálculo de extensões

As regras e especificações de *Sleuth* nos permitem trabalhar com uma aplicação em vários níveis, especificando desde o processo de gerar instâncias de problemas de forma similar à que faz a ferramente do site, até propôr uma estratégia de Inteligência Artificial para administrar os recursos de investigação e manter os dados à parte. Nosso interesse neste trabalho, no entanto, é de ilustrar o uso da LPR como meio de modelagem da aplicação para a tomada de decisões, o que neste caso, é decidir, a partir das informações coletadas, a quem acusar. Dentre todos os aspectos que podem ser abordados neste jogo, o principal deles para a escolha de um suspeito a quem acusar é o conjunto de dados que podem ser considerados na investigação, e como pretendemos utilizar regras em lógica de primeira-ordem, precisamos restringir os aspectos abordados no jogo em termos da expressividade necessária para descrever cada coisa. Nesse ínterim, podemos considerar os seguintes aspectos junto à sua relevância para nossa proposta e os recursos necessários para expressá-los logicamente:

- As perguntas a serem feitas a cada NPC e suspeito são bastante limitadas, podendo variar entre nenhuma pergunta e três (com exceção dos contatos conquistados), e sugerem utilização de contagem para administrar tais recursos. Uma vez que a contagem é recurso que exige expressividade aquém da lógica de primeira-ordem, desconsideraremos esse fator em função de simplificar a aplicação e torná-la melhor de entender e trabalhar. Poderíamos expressar as perguntas que podem ser feitas, bem como qualquer outro aspecto dessa natureza em termos de uma variedade de predicados (como pergunta1(X), pergunta2(X), pergunta3(X), verdadeiros se fosse possível realizar a pergunta), mas isso exigiria um número grande de alterações para gerenciar esses predicados. Além disso, tal tratamento seria feito em função de um fator desinteressante à nossa proposta, uma vez que buscamos apenas racionalizar sobre as informações obtidas em função de determinar se o assassino já pode ser determinado;

- As consultas a NPC's para verificação de pertinência das evidências aos suspeitos é realizada através de perguntas as NPC's e se inclui no caso anterior. Assim sendo, uma vez que somente nos interessa considerar se é o caso que as evidências pertencem a cada suspeito, descartamos a necessidade de verificar durante o cálculo se há uma combinação prova-suspeito, até mesmo porque isso demandaria conhecimento prévio dos dados do problema, o que também está fora do escopo da aplicação. Essa funcionalidade, assim como a anterior seria particularmente interessante caso pudéssemos gerar os casos aleatóriamente e permitir que as entrevistas fossem realizadas. Em outras palavras, somente se fôssemos recriar o jogo e embutir um agente de IA capaz de resolver casos;
- As ações e informações relevantes para que consigamos favores, contatos e status junto às facções sugere um conjunto de predicados especiais para lidar com essas características. O status junto às facções, porém, é discretizado em termos numéricos, podendo ser positivo ou negativo, e também tornaria necessária a utilização do princípio de contagem. Uma vez que contatos e facções só relevantes em termos de carreira do detetive ou para as entrevista de contatos, essas informações são desnecessárias durante cada resolução de caso, onde se encaixa nossa proposta. Assim sendo, desconsideraremos as informações quanto a adquirir contatos e status com facções;
- As possíveis habilidades a serem adquiridas no jogo são tais que demandariam tratamento especial, e grande parte delas serve somente para interferir diretamente nas chances de persuadir os entrevistados. Uma vez que abolimos modelagem de entrevistas, descartamos a necessidade de administrar várias habilidades, e proporemos um conjunto de habilidades interessantes ao detetive para o ambiente dos casos propostos a nós (Assim como o próprio jogo faz, pois a cada novo caso, o detetive já tem um conjunto de habilidades pré-determinado);
- Itens de equipamento conseguidos em missões e comprados na loja de detetives servem para melhorar os valores nos nossos atributos e influenciam nas porcentagens que temos de conseguir persuadir os suspeitos ou determinar características das provas. Como estaremos lidando com informações que foram conseguidas durante o processo de investigação, pouco importa os valores de atributos que tivermos ou itens de equipamento que carregarmos e porcentagens que tínhamos de suceder junto à persuasão de um suspeito ou avaliação de uma evidência. O que

nos importa é se conseguimos ou não. Essas informações seriam relevantes em caso de querermos criar uma ferramenta que gere os dados do caso aleatoriamente. As porcentagens sugerem outro recurso de expressividade que não é presente na lógica de primeira ordem, sugerindo lógica bayesiana ou fuzzy para tratar com esses aspectos, mais uma vez fugindo dos limites encontrados em primeira ordem.

- As regras do jogo quanto às informações já obtidas em investigação determinam condições para a conclusão de certas ocorrências de predicados em elementos do domínio, bem como a possibilidade de verificar o valor verdade dos mesmos sobre qualquer valor. Essas regras determinam grande parte dos passos necessários para detalharmos as regras de inferência lógicas que compõem o sistema de prova para *Sleuth*, e são indispensáveis, portanto. Verificaremos que todas essas regras podem ser perfeitamente escritas dentro de lógica de primeira-ordem, e se fosse o caso de haverem exceções dessa especificação, teríamos de contornar de alguma forma, pois tais regras são realmente importantes para o funcionamento do jogo e a dedução em si, regendo tanto restrições para a automatização da criação de casos aleatórios como direcionando as conclusões que podem ser obtidas através da observação de dados do caso.
- As caraterísticas pertinentes a cada suspeito e provas refletem diferenciação entre os suspeitos e regem possibilidades de combinação prova-suspeito, o que pode ser determinado de formas alheias à consulta a NPC's. Assim sendo manteremos e administraremos esses conceitos na forma de novos predicados, de forma que estes modelam as informações de maior variação entre os casos e são referenciadas em regras muito importantes para a conclusão de um caso.

Assim sendo, devemos considerar nossa aplicação sobre um par de domínios que serão referidos pelos predicados a serem propostos. Após as especificações acima, é notória a necessidade de separar os elementos desses dois conjuntos em domínios diferentes, pois teremos predicados próprios para trabalhar cada um deles, ou mesmo elementos de cada um dos conjuntos simultaneamente. Definimos abaixo os domínios:

- S = {Carlile, Dorothy, Leland, Scott, Zoe, Chancey, Barnabas, Rose, Susan} é o domínio dos suspeitos no caso que desejamos resolver;
- P = {Escrita, Cabelo1, Cabelo2, Pegada} é o domínio das evidências ou provas encontradas na cena do crime.

Esses nomes de suspeitos e referências às evidências encontradas serão utilizados como constantes em função de aplicarmos os predicados definidos e permitir associar uma séria de informações em função de chegar à conclusão de quem é o autor do crime. Muito embora os nomes já sugiram o que foi encontrado na cena do crime, utilizaremos predicados apropriados para representar suas características particulares e definir o tipo de cada prova de acordo com o que citamos previamente como possibilidades. A seguir, definimos algumas regras do jogo de forma mais próxima do que desejamos modelar em lógica, em virtude de um melhor entendimento dos passos a seguir e como justificativa de alguns dos predicados sugeridos:

- Existe somente um assassino entre os suspeitos.
- É garantido que existe um assassino e que este sempre está entre os suspeitos (Desde que todos sejam encontrados, o que consideraremos verdadeiro).
- O assassino não tem um álibi OU dá um álibi falso (Ambos os casos considerados falsos como simplificação).
- Ausência de provas da presença de alguém na cena do crime determina a inocência desta pessoa.
- Um álibi falso com evidência física comprovada determina culpa.
- Um álibi falso com testemunho simples determina culpa.
- Uma evidência física comprovada com duplo testemunho determina culpa.
- Um álibi verdadeiro determina inocência.s
- Através de uma pesquisa, pode-se determinar a inocência de uma pessoa (Limitado a uma pessoa).
- Um suspeito morto durante a investigação é considerado inocente do crime investigado.
- Todas as evidências físicas encontradas são relacionadas aos suspeitos.
- Cada evidência física é relacionada a um suspeito diferente e a somente um suspeito.
- Os testemunhos podem não ser sobre o verdadeiro culpado (São opiniões).
- Uma prova que indica pessoa canhota só pode ter sido produzida por um suspeito canhoto, assim como ocorre de forma similar para todas as características de suspeitos e provas no que se refere à combinação dos mesmos (destro, cabelos crespos ou lisos, homem ou mulher, pesado ou não).

Observações sobre o contexto suposto pelo conjunto de regras acima:

- Uma evidência comprovada com testemunho simples NÃO determina culpa (Forte indício).

- Normalmente, um suspeito com álibi não verificado, mas que tem prova de sua presença no local e algum testemunho contra, é o culpado.
- Um suspeito não pode ser simultaneamente inocente e assassino.
- Um álibi pode não ter sido comprovado verdadeiro nem falso, embora não possa ser simultaneamente verdadeiro e falso.
- Para que o álibi seja verdadeiro OU falso, ele precisa primeiro ser descoberto.
- Normalmente, somente o assassino leva *mais de uma* pessoa a testemunhar contra ele.

Agora, com um conjunto já definido de regras para o jogo, podemos começar a extrair predicados e determinar o que cada um representa e sobre quais domínios cada um será aplicável. Mantemos em mente que esta é uma lista de predicados de acordo com o que será necessário mais à frente para a especificação destas regras em um programa prolog sem negação, portanto introduzindo predicados que podem parecer ser, em primeira vista, desnecessários. Seguimos com uma listagem dos mesmos, utilizando, por convenção, as variáveis X e Y para suspeitos e P para provas. Por estas variáveis, nos referenciaremos a suspeitos e provas em qualquer ponto do texto daqui em diante.
Predicados específicos sobre suspeitos:

- assassino(X):
  Indica se o indivíduo é o assassino quando verdadeiro. Seu valor falso indica simplesmente que não se pode garantir que é verdade OU que o elemento X é considerado inocente.
- inocente(X):
  Indica se o indivíduo foi inocentado pelas evidências quando verdadeiro. Falso indica simplesmente que não se pode garantir que é verdade OU que o elemento X é considerado culpado.
- álibi(X):
  Quando verdadeiro, indica que o álibi do suspeito X é conhecido.
- verdadeiro(X):
  O álibi do suspeito X foi confirmado, se verdadeiro. Caso contrário, ou falta confirmação ou o mesmo foi comprovado falso.
- falso(X):
  O álibi do suspeito X foi derrubado, se verdadeiro. Caso contrário, ou falta confirmação ou o mesmo foi comprovado verdadeiro.
- prova(X):
  Uma das evidências físicas coletadas na cena do crime está relacionada ao suspeito X, se verdadeiro.

- testemunho(X):
  Alguém desconfia fortemente do suspeito X e testemunha contra ele. Não é suficiente para culpar alguém, pois alguém pode desconfiar de um inocente.
- duplotestemunho(X):
  Mais de uma pessoa desconfia fortemente do suspeito X e testemunha contra ele. É suficiente para identificar o assassino.
- pesquisa(X):
  Através de pesquisa, determina-se que o suspeito X estava compromissado ou ocupado de alguma forma no momento ou dia do crime. Essa pesquisa derruba um álibi falso, e comprova inocência, pois o suspeito o teria utilizado somente para não chamar atenção à sua atividade.
- morto(X):
  O suspeito teria sido morto durante as investigações, pelo mesmo assassino que é investigado, com propósito de eliminar testemunha ou por sequência premeditada.
- slim(X):
  Indica que o suspeito X apresenta constituição corporal leve.
- heavy(X):
  Indica que o suspeito X apresenta constituição corporal pesada.
- straight(X):
  Indica que o suspeito X apresenta cabelos lisos.
- curly(X):
  Indica que o suspeito X apresenta cabelos crespos.
- woman(X):
  Indica que o suspeito X é do sexo feminino.
- man(X):
  Indica que o suspeito X é do sexo masculino.
- right(X):
  Indica que o suspeito X é destro.
- left(X):
  Indica que o suspeito X é canhoto.
- testemunho(X):
  Indica que algum suspeito apresenta testemunho contra X.
- duplotestemunho(X):
  Indica que dois suspeitos apresentam testemunhos contra X.
- prova(X):
  Quando verdadeiro, significa que alguma prova foi associada ao suspeito X.
- desfecho(X):
  Ocorre verdadeiro quando dois elementos que indicam culpa de X são

encontrados. É suficiente para determinar que X é o assassino.

- quaseconclusao(X):
  Caso não tenhamos um desfecho no caso, podemos recorrer à quase conclusão. Este predicado é verdadeiro sobre X, caso exista uma peça de evidência física confirmada contra o suspeito X e um testemunho simples.
- evidencia(X):
  É verdade quando temos ao menos uma peça de culpa definitiva entre álibi falso, prova ou testemunho contra o suspeito X.
- diferente(X, Y): Garante que os suspeitos X e Y sejam vistos como pessoas diferentes em alguns momentos das provas onde essa informação é essencial.

A seguir, introduzimos alguns predicados mais que regem uma "hierarquia de culpa", de forma que a cada novo predicado destes que é considerado verdadeiro para algum suspeito, estamos, então, mais próximos da conclusão de um caso.

- suspeito_assassino(X):
  Este predicado é verdadeiro quando temos alguma evidência de que o suspeito X pode ser o assassino, ou seja, quando encontramos a primeira peça de evidência para comprovar culpa de um suspeito.
- evidente_assassino(X):
  Verificaremos a verdade deste predicado aplicado a algum suspeito X para o qual estejamos próximos a uma conclusão, de acordo com as especificações do predicado quase_conclusão(X).
- provavel_assassino(X):
  É mais forte que a afirmação de assassino evidente do predicado anterior, pois ocorre quando um assassino evidente não tem inocência comprovada, ou seja, quando a prova de inocência falha. Essa é nossa melhor aposta caso não conheçamos prova concreta para acusar a nenhum suspeito. Qualquer conjunto de afirmações mais forte que esta é suficiente para concluir a culpa de alguém, e são sugeridas como provas do predicado assassino(X). Esse outro predicado é importante, pois pode ser utilizado como resposta, enquanto que o anterior não pode, sendo apenas um passo intermediário.

Os predicados criados para manipular características das provas, são:

- nota(P):
  Determina que a prova P é uma nota escrita, e serve para análise de caligrafia para determinar um possível assassino.

- hair(P):
  Determina que a prova P é um fio de cabelo, e serve para comparação dos fios com os cabelos dos suspeitos para determinar um possível assassino.
- shoe(P):
  Determina que a prova P é uma pegada, e serve para análise da sola dos sapatos dos suspeitos para determinar um possível assassino.
- sexo(P):
  Determina que a prova P é uma fibra de tecido de roupas, e serve para análise compatativa com as roupas dos suspeitos para determinar um possível assassino.
- destro(P):
  Aplica-se somente sobre provas do tipo nota, e determina que a escrita é de um suspeito destro. Somente suspeitos tais que right(X) é verdade podem combinar com uma prova tal que destro(P) é verdade.
- canhoto(P):
  Aplica-se somente sobre provas do tipo nota, e determina que a escrita é de um suspeito canhoto. Somente suspeitos tais que left(X) é verdade podem combinar com uma prova tal que canhoto(P) é verdade.
- liso(P):
  Aplica-se somente sobre provas do tipo cabelo, e determina que os fios são de um suspeito de cabelos lisos. Somente suspeitos tais que straight(X) é verdade podem combinar com uma prova tal que liso(P) é verdade.
- crespo(P):
  Aplica-se somente sobre provas do tipo cabelo, e determina que os fios são de um suspeito de cabelos crespos. Somente suspeitos tais que curly(X) é verdade podem combinar com uma prova tal que crespo(P) é verdade.
- leve(P):
  Aplica-se somente sobre provas do tipo pegada, e determina que a evidência encontrada é de um suspeito de constituição leve. Somente suspeitos tais que slim(X) é verdade podem combinar com uma prova tal que leve(P) é verdade.
- pesado(P):
  Aplica-se somente sobre provas do tipo pegada, e determina que a evidência encontrada é de um suspeito de constituição pesada. Somente suspeitos tais que heavy(X) é verdade podem combinar com uma prova tal que pesado(P) é verdade.
- mulher(P):
  Aplica-se somente sobre provas do tipo sexo, e determina que as fibras

de tecido encontradas são de um suspeito do sexo feminino. Somente suspeitos tais que woman(X) é verdade podem combinar com uma prova tal que mulher(P) é verdade.

- homem(P):
  Aplica-se somente sobre provas do tipo sexo, e determina que as fibras de tecido encontradas são de um suspeito do sexo masculino. Somente suspeitos tais que man(X) é verdade podem combinar com uma prova tal que homem(P) é verdade.

Finalmente, apresentamos os predicados que são utilizados para relacionar suspeitos e provas e são aplicados aos dois domínios simultâneamente:

- match(P, X):
  Este predicado é verdadeiro quando descobrimos que a prova P foi produzida pelo suspeito X.
- nao_match(P, X):
  Este predicado é verdadeiro quando podemos garantir que a prova P *não* foi produzida pelo suspeito X.

As características de cada suspeito ou prova são cobertos através destes predicados, de forma que uma nota P_1 escrita com caligrafia de uma pessoa canhota é representada pelas ocorrências de nota(P_1) e canhoto(P_1); Similarmente, temos que um suspeito X_1 homem, destro, pesado e de cabelos crespos é representado pela ocorrência das cláusulas man(X_1), right(X_1), heavy(X_1) e curly(X_1) no modelo. Além disso, podemos demonstrar que não é verdade que P_1 é uma evidência produzida por X_1 e representar essa informação com a cláusula nao_match(P_1, X_1).
Para permitir uma demonstração mais didática e um conjunto de predicados e regras LPR mais conciso e legível, temos feito uma série de simplificações. Até então, grande parte das regras do jogo sugere uma cláusula lógica direta e um conjunto que pode ser trabalhado dedutivamente, exceto por algumas das observações. Nesse sentido, o que ocorre é que algumas especificações são obtidas através da experiência com o jogo, e da observação de aspectos variantes entre casos diferentes. De certa forma, quando utilizamos essas especificações, coisas que ocorrem comumente podem ser tomadas como regras fixas em virtude de complementar o quadro de regras do modelo e reger a investigação. É o que acontece em investigações reais quando os detetives utilizam-se, de forma muito similar, de sua experiência e intuição. Esses fatores são levados em conta para determinar como seguir a investigação quando os fatos apontam para múltiplos caminhos, ou mesmo quando ao fim de todos os passos é necessário fazer uma última escolha.

Em *Sleuth* percebemos esses fatores com a prática, e tanto notar essas regras implícitas (sim, pois tudo que rege o jogo *são* regras, uma vez da produção dos casos como explicada anteriormente) quanto utilizá-las sugere exatamente uma melhor capacidade de resolver casos através de inteligência, experiência ou intuição. Essas regras vão normalmente aparecer como generalizações no nosso modelo, uma vez que decisões tomadas dessa forma normalmente seguem uma tendência ampliativa. Seriam decisões tomadas, muitas vezes, de acordo com uma direção ligeiramente diferente da qual apontam os fatos, muito embora, em nosso modelo, representem apenas informações implícitas e, portanto, não sugeridas pelos fatos, constituindo questões pertinentes à prática da "profissão".

Para complementar a caracterização da porção que modelamos nesta aplicação, destacamos alguns outros fatores que serão considerados em nossa investigação.
Em função de organizar as regras sem a necessidade de novos predicados, supomos simplesmente que foi possível, através de entrevistas, descobrir todos os suspeitos no caso. Essa característica quase sempre ocorre em jogos de dificuldade mais baixa, e no caso do nosso modelo e dos cálculos inferenciais realizados para descobrir o assassino, poderíamos simplesmente introduzir um elemento no domínio de suspeitos chamado "não encontrado". Uma vez que os suspeitos não encontrados não teriam características físicas conhecida, nem poderíamos determinar se as provas pertendem ou não a essas pessoas, seria simplesmente impossível determinar match(não encontrado) ou nao_match(não encontrado), e consequentemente provar match(X) para qualquer suspeito por exclusão, ou seja, nenhuma mudança precisaria ser feita no modelo, pois a funcionalidade dos predicados propostos independe disso. Assim, o assassino é obrigatoriamente um elemento do conjunto de suspeitos, mesmo que seja o elemento "não encontrado". No nosso exemplo inicial, como já determinamos, temos encontrado os nove suspeitos (essa forma ilustra melhor as provas).

Começaremos considerando todos possivelmente culpados (não inocentes), mesmo que não consideremos todos como assassinos. Essa é uma ação intuitiva no sentido de que todos são inocentes até que se prove o contrário, mas podendo ser o culpado qualquer um dos suspeitos, cada um deve ter sua inocência comprovada inicialmente. Uma das formas de encontrar o culpado é, portanto, por exclusão dos inocentes, comprovando a inocência de cada um até que só reste um suspeito, muito embora essa não seja a forma mais comum, pois seriam muitos suspeitos pra provas inocência. Ainda assim, eliminar suspeitos desta forma é eliminar opções e estreitar a abrangência da investigação, tornando essa uma ferramenta poderosa e importante. Ressaltamos que somente quatro pessoas podem ser realmente culpadas, que são aqueles suspeitos a quem pertence alguma peça de evidência encontrada na cena do crime. Com isso em mente, considerando simplesmente que oito em nove suspeitos são inocentes, podemos dizer que, em geral, os suspeitos são inocentes, e que a exceção é o assassino e modelar essa regra como generalização.

Sobre testemunhos simples, devemos considerar que nem sempre esse testemunho realmente está ligado ao assassino, muito embora seja uma forte chance caso tenhamos mais evidências de sua culpa. Se a outra evidência for a comprovação de que seu álibi é falso, é o bastante para acusar o assassino. Isso ocorre porque normalmente haverá testemunho contra algum suspeito inocente, mas o assassino *sempre* tem ao menos duas pessoas que sabem de algo com o qual podem testemunhar contra este. Em geral, os inocentes só têm realmente uma pessoa que testemunhe contra.

Uma vez que não modelaremos as entrevistas e recursos disponíveis de perguntas, os álibis não são anotados no modelo. Somente nos interessa descobrir se cada suspeito tem um álibi e se este é verdadeiro ou falso. Assim sendo, quando o suspeito responder à pergunta de álibi alegando não ter nenhum, o consideraremos automaticamente falso por convenção do jogo.

Ressaltamos, ainda, que ao encontrar-nos com um suspeito, sua descrição física e características são automaticamente avaliadas e reconhecidas para comparação com as evidências da cena do crime (Veja a figura 4). Uma vez que sugerimos ter encontrado todos os suspeitos (mesmo que através de um suspeito nulo "não encontrado"), sempre conhecemos as características físicas de todos os suspeitos.

.

Feitas as considerações e esclarecimentos necessários à especificação de conceitos sobre o jogo e nosso modelo, podemos começar a especificar os predicados e regras que farão parte deste. O conjunto de regras que utilizaremos em nossa aplicação-exemplo é dividido em duas partes por conveniência de organização. Uma vez que precisamos introduzir os domínios de problema no modelo para podermos definir o culpado, necessitaremos de um conjunto de regras especificadas sobre os mesmo, detalhando as características de cada suspeito e prova e todas as informações, como previamente citado. Essa parte das regras são variantes de acordo com o caso a ser explorado, e deve ser considerado à parte, uma vez que as regras restantes especificam o comporta-

08 - Entrevista de suspeitos.png

mento dos predicados e as consequências que qualquer grupo de informações sugira no sentido de chegarmos a uma conclusão. Assim sendo, apresentaremos inicialmente o modelo que se refere às regras do jogo. Cada regra lógica é especificada a seguir junto à regra escrita em linguagem natural como a interpretamos.

- $(assassino(X) \wedge assassino(Y)) \rightarrow (X = Y)$;
  Somente existe um assassino.
- $\exists X(assassino(X))$ -(;
  Em geral, existe um assassino (É verdade, desde que encontremos a todos os suspeitos).
- $verdadeiro(X) \rightarrow inocente(X)$;
  Álibis verdadeiros indicam inocência.
- $\forall P(\neg match(P, X)) \rightarrow inocente(X)$;
  Se podemos provar que nenhuma prova bate com o suspeito X, então X é inocente.
- $pesquisa(X) \rightarrow inocente(X)$;
  Quando a pesquisa realizada nos traz algum acontecimento envolvendo o suspeito X, este é inocente.
- $morto(X) \rightarrow inocente(X)$;
  Personagens mortos durante a investigação são considerados inocentes.
- $\forall P \exists X match(P, X)$;

Para toda prova, existe um suspeito com o qual podemos encontrar uma combinação prova-suspeito.

- $\forall X(match(P, X) \wedge match(Q, X) \rightarrow (P = Q))$;

  Somente uma evidência é tal que encontramos combinação prova-suspeito com o suspeito X.
- $\forall P(match(P, X) \wedge match(P, Y) \rightarrow (X = Y))$;

  Somente um suspeito é tal que encontramos combinação prova-suspeito com a evidência P.
- $\exists P((match(P, X) \wedge assassino(X))$;

  Existe uma prova que pode ser posta em combinação prova-suspeito com o assassino.
- $\neg verdadeiro(X) \wedge testemunho(X) \rightarrow desfecho(X)$;

  Um álibi falso e testemunho simples levam a uma conclusão do caso.
- $\neg verdadeiro(X) \wedge prova(X) \rightarrow desfecho(X)$;

  Um álibi falso e combinação prova-suspeito levam a uma conclusão do caso.
- $duplotestemunho(X) \rightarrow desfecho(X)$ *-(* ;

  Em geral, o testemunho duplo contra um mesmo suspeito leva a uma conclusão do caso.
- $\exists X \forall Y((X \neq Y) \rightarrow inocente(Y)) \rightarrow desfecho(X)$
- $inocente(X) \leftrightarrow \neg assassino(X)$;

  Os predicados inocente e assassino têm sentidos opostos.
- $(pesquisa(X) \wedge pesquisa(Y)) \rightarrow (X = Y)$;

  Somente pode ser encontrada evidência de inocência por pesquisa para um suspeito.
- $(verdadeiro(X) \vee falso(X)) \rightarrow alibi(X)$;

  Somente é possível aplicar os predicados verdadeiro(X) e falso(X) a suspeitos que tenham nos fornecidos algum álibi.
- $destro(P) \rightarrow (\neg right(X) \rightarrow \neg match(P, X))$;

  Se P é uma evidência produzida por uma pessoa destra, então se X não é destro, P não foi produzida por X.
- $\neg destro(P) \rightarrow (right(X) \rightarrow \neg match(P, X))$;

  Se P é uma evidência produzida por uma pessoa canhota (não destra), então se X é destro, P não foi produzida por X.
- $liso(P) \rightarrow (\neg straight(X) \rightarrow \neg match(P, X))$;

  Se P é uma evidência produzida por uma pessoa de cabelos lisos, então se X não tem cabelos lisos, P não foi produzida por X.
- $\neg liso(P) \rightarrow (straight(X) \rightarrow \neg match(P, X))$;

  Se P é uma evidência produzida por uma pessoa de cabelos crespos (não lisos), então se X tem cabelos lisos, P não foi produzida por X.

- $leve(P) \rightarrow (\neg slim(X) \rightarrow \neg match(P, X));$

  Se P é uma evidência produzida por uma pessoa leve, então se X não é leve, P não foi produzida por X.
- $\neg leve(P) \rightarrow (slim(X) \rightarrow \neg match(P, X));$

  Se P é uma evidência produzida por uma pessoa pesada (não leve), então se X é leve, P não foi produzida por X.
- $homem(P) \rightarrow (\neg man(X) \rightarrow \neg match(P, X));$

  Se P é uma evidência produzida por um homem, então se X não é homem, P não foi produzida por X.
- $\neg homem(P) \rightarrow (man(X) \rightarrow \neg match(P, X));$

  Se P é uma evidência produzida por uma mulher (não homem), então se X é homem, P não foi produzida por X.
- $\neg verdadeiro(X) \lor duplotestemunho(X) \lor prova(X) \rightarrow evidencia(X);$

  Álibi falso, testemunho duplo contra o suspeito ou prova de sua presença no crime constituem evidência contra ele.
- $prova(X) \land testemunho(X) \rightarrow quaseconclusao(X);$

  Prova da presença do suspeito na cena do crime e testemunho contra ele é forte indício de culpa, ou seja, estamos quase em uma conclusão do caso.
- $prova(X) \rightarrow evidencia(X);$

  Constatação de que uma evidência física da cena do crime foi produzida por um suspeito é evidência de um possível envolvimento no crime por parte deste.
- $testemunho(X) \rightarrow evidencia(X);$

  Algum testemunho contra qualquer suspeito é evidência de um possível envolvimento no crime por parte deste.
- $falso(X) \rightarrow evidencia(X);$

  Um álibi dado por um suspeito que for constatado falso é evidência de um possível envolvimento no crime por parte deste.
- $evidencia(X) \land suspeito(X) \rightarrow suspeito\_assassino(X);$

  Um suspeito que constatamos ter alguma peça de evidência contra si pode mais facilmente estar ligado ao assassinato.
- $quase\_conclusao(X) \land suspeito(X) \rightarrow possivel\_assassino(X);$

  Quando nos aproximamos de uma conclusão sobre a culpa de qualquer suspeito, devemos considerá-lo como uma grande possibilidade de ser o assassino.
- $possivel\_assassino(X) \land \neg alibi(X) \rightarrow provavel\_assassino(X);$

  Um possível assassino para o qual não pudemos verificar se existe álibi, tem maior probabilidade do que alguém que deu um álibi que não pôde ser verificado.
- $suspeito(X) \land desfecho(X) \rightarrow assassino(X);$

Se tivermos informações suficientes para um desfecho, então temos o assassino.

- $suspeito(X)$ -($inocente(X)$;

  Todos são suspeitos, à exceção de quando provamos sua inocência.
- $\exists X(testemunho(X) \wedge inocente(X))$ -(;

  Em geral, ocorrem testemunhos contra pessoas inocentes.
- $provavel\_assassino(X) \rightarrow assassino(X)$ -($\exists Y desfecho(Y)$;

  Normalmente, o suspeito que for encontrado com indício o bastante para ser considerado o provável assassino é o culpado, à exceção de conseguirmos um desfecho definitivo para o caso.
- $possivel\_assassino(X) \rightarrow assassino(X)$ -($\exists Y provavel\_assassino(Y)$;

  Em geral, acusar um possível assassino é seguro, a não ser que tenhamos indício mais forte sobre outro suspeito.
- $suspeito\_assassino(X) \rightarrow assassino(X)$ -($\exists Y possivel\_assassino(Y)$;

  Suspeitos que tenham qualquer evidência podem ser acusados, a menos que não tenhamos nenhum indício mais forte da autoria por outro suspeito.
- $suspeito(X) \rightarrow assassino(X)$ -($\exists Y suspeito\_assassino(Y)$;

  Quaisquer suspeitos podem vir a ser acusados, caso não seja possível encontramos absolutamente nenhuma evidência contra estes.

As regras acima, especificam tanto limites de aplicação dos predicados quando os casos são gerados quanto regras propriamente como são apresentas formalmente aos jogadores. Uma vez que as especificações orientadas à criação de novos casos têm intuito de definir a forma como pode ocorrer o crime e acrescentar algum bom senso impondo limites ao que seriam as possibilidades de os suspeitos aparentarem estar, individualemente, mais ou menos envolvidos nos crimes, podemos utilizar essas informações também a nosso favor. No mesmo sentido, como bom senso, sabemos e utilizamos informações de que somente há um assassino, somente uma pessoa pode ser relacionada a cada prova e etc. Quando criarmos uma base de informações teórica para um suposto caso que desejemos resolver, precisamos seguir essas orientações ao definir como os predicados vão se aplicar aos elementos de cada domínio. A partir destas, definimos a teoria-LPR inicial $\tau_0 = \langle T_0, G_0 \rangle$, de forma que T_0 é o conjunto de regras, dentre estas acima definidas, tais que essas regras *não são* generalizações, e G_0 é o conjunto de regras, dentre estas acima definidas, tais que essas regras *são* generalizações. Umas vez que, como comprovaremos, durante o processo que executaremos, não haverá variações em G_0, seguiremos tratando o conjunto de generalizações simplesmente por G. A teoria inicial τ_0 será considerada em todos os casos, e comporá a base-LPR desde o começo do processo.

Como ocorreria em uma investigação no jogo ou na vida real, as pistas e informações são descobertas gradativamente e vão compondo um conjunto dedutivo junto a suposições e expectativas. Assim sugere a utilização da LPR para modelagem, e nesse sentido trabalham nossas regras especificadas sob a forma de generalizações. Ressaltamos que, em primeira vista, o modelo apresenta poucas regras de generalização, mas a forma como as generalizações proposta que têm exceção são revistas a cada nova informação descoberta, uma vez que estas fazem parte do modelo para direcionar a descoberta do culpado. Como objetivo único em nossos cálculos de calcular a extensão LPR da teoria proposta para o jogo *Sleuth*, vamos ilustrar o cálculo de alguns passos do processo realizado em busca de um assassino. Apresentaremos um conjunto de informações que compõe tudo que é descoberto durante as entrevistas e verificações de álibis, e etc. Em seguida, apresentaremos uma estratégia de sequência do jogo para possibilitar a descoberta dessas informações. A cada vez que mudamos qualquer coisa nesse conjunto de informações, é necessário verificar se algo mudou na extensão LPR, checando se a adição dessas novas informações provoca a prova da exceção de alguma generalização, o que equivale, em verdade, a refazer todo o cálculo. Dessa forma, estaremos constantemente refazendo o cálculo da extensão e demonstrando em alguns pontos da execução o que pode ser provado. Podemos verificar variações na teoria envolvendo quaisquer predicados que apareçam em generalizações.

Como estratégia de investigação, sugerindo uma forma de administrar os recursos fora deste framework lógico proposto pela modelagem, propomos os seguinte passos, em ordem:

- Consultar contatos sobre rumores (pois não gasta recursos) e observar a cena do crime.
- Perguntar a suspeitos que não possam ter produzido nenhuma prova e pareçam dispostos a responder perguntas por outras pessoas que pudessem ter razões para cometer o crime (e anotar as características físicas dessas pessoas).
- Realizar uma pesquisa para tentar reduzir o número de possíveis suspeitos. Caso não seja possível encontrar nada, as ocorrências envolvem algum suspeito que ainda não foi encontrado, o que sugere que tentemos, de novo caso mais suspeitos sejam encontrados.
- Seguir perguntando aos suspeitos mais dispostos por álibis e outras pessoas com razões, tendo em mente quem são os suspeitos com mais

ou menos chances de terem produzidos provas. Nesse caso, devemos priorizar descobrir álibis e confirmá-los quando o suspeito tem possibilidade maior de ter deixado alguma prova, e quando os álibis envolvem NPC's que pareçam dispostas a responder um maior número de perguntas.

- Verificar os álibis das pessoas com maior chance de serem suspeitos junto aos NPC's.
- Buscar descobrir quem são os suspeitos que deixaram as provas na cena do crime. Tanto combinações com suspeitos quanto com inocentes são úteis, pois criam possibilidades para descobrirmos qual é a prova deixada pelo assassino.
- Utilizar as últimas perguntas para tentar descobrir sobre testemunhos contra outros suspeitos.

Essa estratégia parece bastante eficaz e será utilizada no nosso exemplo. Detalhes como a fonte de cada informação são irrelevantes para nossos cálculos, de forma que os passos acima são adotados para sugerir uma sequência na qual as informações acima serão acrescentadas pouco a pouco ao nosso modelo de regras para verificarmos como realizar o cálculo. Assim sendo, consideraremos primeiramente o começo do caso como incluindo em nosso modelo poucas cláusulas e verificaremos a extensão gerada nesse momento. Prosseguiremos acrescentando cláusulas de acordo com certos passos e mostrando como a teoria monotônica de informações é formada a cada passo, através de inclusão e substituição de algumas regras. Os dados daí extraídos são inclusos na parte monotônica da teoria LPR e servirão como base para as inferências ampliativas sugeridas pelas generalizações apresentadas entre as regras acima.

A título de referência, agrupamos a seguir as generalizações avaliadas e as enumeramos para facilitar o acompanhamento dos passos que seguiremos quando considerando se cada generalização entra na extensão.

- $G_1 = \exists X(assassino(X))$ -(;
- $G_2 = duplotestemunho(X) \rightarrow desfecho(X)$ -(;
- $G_3 = suspeito(X)$ -($inocente(X)$;
- $G_4 = \exists X(testemunho(X) \wedge inocente(X))$ -(;
- $G_5 = provavel\_assassino(X) \rightarrow assassino(X)$ -($\exists Y desfecho(Y)$;
- $G_6 = possivel\_assassino(X) \rightarrow assassino(X)$ -($\exists Y provavel\_assassino(Y)$;
- $G_7 = suspeito\_assassino(X) \rightarrow assassino(X)$ -($\exists Y possivel\_assassino(Y)$;
- $G_8 = suspeito(X) \rightarrow assassino(X)$ -($\exists Y suspeito\_assassino(Y)$;

Para efeitos do cálculo que segue, consideramos cada uma dessas regras como representante da coleção de instâncias obtidas ao substituir a variável X por alguma constante do conjunto de pessoas definido para este caso. Por exemplo, a regra G_3 é a coleção das instâncias:

- $suspeito(carlile)$ -($inocente(carlile)$;
- $suspeito(dorothy)$ -($inocente(dorothy)$;
- $suspeito(leland)$ -($inocente(leland)$;
- $suspeito(barnabas)$ -($inocente(barnabas)$;
- $suspeito(rose)$ -($inocente(rose)$;
- $suspeito(susan)$ -($inocente(susan)$;
- $suspeito(zoe)$ -($inocente(zoe)$;
- $suspeito(scott)$ -($inocente(scott)$;
- $suspeito(chancey)$ -($inocente(chancey)$;
- $\forall X suspeito(X)$ -($\exists X inocente(X)$.

A última instância é, em verdade, uma junção dos resultados, representando que, caso não tenhamos nenhuma exceção, em nenhum das instâncias, provada, então podemos utilizar a conclusão desta regra com uma aplicação do quantificador universal.

Reunimos essas instâncias na regra descrita G_3 para uma descrição simplificada das verificações de cada instância quando verificando se é o caso de suas exceções serem provadas pelo conjunto parcial da extensão quando utilizando o operador $\Psi\tau(\Phi)$. Descartamos, nesse sentido, comentários sobre a última instância em cada momento e verificamos somente as instâncias que são aplicadas a algum valor de X.

Faz-se importante ressaltar que o nosso foco é mostrar como as generalizações são aplicadas e negadas no cálculo da extensão em cada momento, e, assim sendo, utilizaremos a mesma sequência utilizada nos exemplos de aplicação fornecidos no trabalho [5]. Inicialmente, definiremos o conjunto de informações como novas fórmulas adicionadas a T em τ, criando um novo τ_i a cada iteração a partir de τ_0 como definido anteriormente. Em seguida, mostraremos a extensão que obtemos através do cálculo e condições sugeridas nas definições 2.31 e 2.32. As passagens da investigação com adição de novas informações são descritas em momentos, como segue. Alguns momentos mais interessantes para ilustrar a forma como generalizações são ministradas são escolhidos com essa intenção. Nos primeiros momentos, utilizaremos de mais detalhe para mostrar o cálculo da extensão, mas nos seguintes, omitimos maiores detalhes para simplificar o acompanhamento das diferenças proporcionadas pelos cálculos em cada novo momento. Seguem abaixo os momentos escolhidos da divisão de um caso exemplo.

M1 O detetive é convocado ao início do caso. Dorothy Smith convida o personagem a trabalhar para ela e descobrir quem assassinou seu querido pai. Ela diz suspeitar de Leland Michaels, por ter uma dívida milionária com seu pai, Rose Smith, uma irmã que há tempos brigava com o falecido pela posse de bens de família, e Scott Bright, advogado da vítima que tinha seu nome na lista de herdeiros.

Desse contexto, concluímos as informações modeladas nas seguintes regras LPR, que são anexadas às previamente definidas para criar um modelo composto para este caso que será investigado, determinando uma teoria $\tau_1 = \langle T_1, G \rangle$, onde T_1 é T_0 acrescida em união com o conjunto de regras cujos elementos são os seguintes:

- $slim(dorothy), \neg man(dorothy), \neg straight(dorothy), right(dorothy)$ (Somente temos esse tipo de conclusões sobre Dorothy, pois foi a única com quem tivemos contato);

- A lista de suspeitos consta de 4 elementos, agora. Estes, devem ser visitados para podermos começar a afirmar qualquer coisa sobre a inocência dos mesmos.

Para cada momento do processo que detalhamos, acrescentaremos novas regras para compor um novo T_i e, consequentemente, um novo τ_i. A seguir, calculamos a extensão através do operador $\Psi_{\tau_i}(\Phi)$:

- $T \subseteq \Psi_\tau(\Phi)$:
  * T_1 faz parte de $\Psi_{\tau_1}(\Phi)$.

- se $\Psi_\tau(\Phi) \mid\!\!\underline{\;{}_{\text{LPD}}\;} P$, então $P \in \Psi_\tau(\Phi)$:
  * Quaisquer fórmulas que possam ser provadas a partir das fórmulas de $\Psi_{\tau_1}(\Phi)$, fazem parte de $\Psi_{\tau_1}(\Phi)$.

- se $\mathrm{Lim}(G') \cap \mathrm{Th}_{\mathrm{LPD}}(\Phi \cup \mathrm{Hyp}(G')) = \emptyset$, então $\mathrm{Hyp}(G') \subseteq \Psi_\tau(\Phi)$ e $G' \in \overline{\Psi}_\tau(\Phi)$:

  * A regra $G_1 = \exists X(assassino(X))$ -(faz parte de $\Psi_{\tau_1}(\Phi)$, pois: $\mathrm{Lim}(G_1) = \emptyset$

$\text{Hyp}(G_1) = \exists X(assassino(X))$
Como consequência, $\text{Lim}(G_1) \cap \text{Th}_{\text{LPD}}(\Phi \cup \text{Hyp}(G_1)) = \emptyset$, e portanto, é verdade que:
$\text{Hyp}(G_1) \subseteq \Psi_{\tau_1}(\Phi)$ e $G_1 \in \overline{\Psi}_{\tau_1}(\Phi)$;

* A regra $G_2 = duplotestemunho(X) \rightarrow desfecho(X)$ -(faz parte de $\Psi_{\tau_1}(\Phi)$, pois:
$\text{Lim}(G_2) = \emptyset$
$\text{Hyp}(G_2) = duplotestemunho(X) \rightarrow desfecho(X)$
Como consequência, $\text{Lim}(G_2) \cap \text{Th}_{\text{LPD}}(\Phi \cup \text{Hyp}(G_2)) = \emptyset$, e portanto, é verdade que:
$\text{Hyp}(G_2) \subseteq \Psi_{\tau_1}(\Phi)$ e $G_2 \in \overline{\Psi}_{\tau_1}(\Phi)$;

* A regra $G_4 = \exists X(testemunho(X) \wedge inocente(X))$ -(faz parte de $\Psi_{\tau_1}(\Phi)$, pois:
$\text{Lim}(G_4) = \emptyset$
$\text{Hyp}(G_4) = \exists X(testemunho(X) \wedge inocente(X))$
Como consequência, $\text{Lim}(G_4) \cap \text{Th}_{\text{LPD}}(\Phi \cup \text{Hyp}(G_4)) = \emptyset$, e portanto, é verdade que:
$\text{Hyp}(G_4) \subseteq \Psi_{\tau_1}(\Phi)$ e $G_4 \in \overline{\Psi}_{\tau_1}(\Phi)$;

* A regra $G_5 = provavel\_assassino(X) \rightarrow assassino(X)$ -($\exists Y desfecho(Y)$ faz parte de $\Psi_{\tau_1}(\Phi)$, pois:
$\text{Lim}(G_5) = \exists Y desfecho(Y)$
$\text{Hyp}(G_5) = provavel\_assassino(X) \rightarrow assassino(X)$
Como consequência, $\text{Lim}(G_5) \cap \text{Th}_{\text{LPD}}(\Phi \cup \text{Hyp}(G_5)) = \emptyset$, e portanto, é verdade que:
$\text{Hyp}(G_5) \subseteq \Psi_{\tau_1}(\Phi)$ e $G_5 \in \overline{\Psi}_{\tau_1}(\Phi)$;

* A regra $G_6 = possivel\_assassino(X) \rightarrow assassino(X)$ -($\exists Y provavel\_assassino(Y)$ faz parte de $\Psi_{\tau_1}(\Phi)$, pois:
$\text{Lim}(G_6) = \exists Y provavel\_assassino(Y)$
$\text{Hyp}(G_6) = possivel\_assassino(X) \rightarrow assassino(X)$
Como consequência, $\text{Lim}(G_6) \cap \text{Th}_{\text{LPD}}(\Phi \cup \text{Hyp}(G_6)) = \emptyset$, e portanto, é verdade que:
$\text{Hyp}(G_6) \subseteq \Psi_{\tau_1}(\Phi)$ e $G_6 \in \overline{\Psi}_{\tau_1}(\Phi)$;

* A regra $G_7 = suspeito\_assassino(X) \rightarrow assassino(X)$ -($\exists Y possivel\_assassino(Y)$ faz parte de $\Psi_{\tau_1}(\Phi)$, pois:
$\text{Lim}(G_7) = \exists Y possivel\_assassino(Y)$
$\text{Hyp}(G_7) = suspeito\_assassino(X) \rightarrow assassino(X)$

Como consequência, $\mathrm{Lim}(G_7) \cap \mathrm{Th}_{\mathrm{LPD}}(\Phi \cup \mathrm{Hyp}(G_7)) = \emptyset$, e portanto, é verdade que:
$\mathrm{Hyp}(G_7) \subseteq \Psi_{\tau_1}(\Phi)$ e $G_7 \in \overline{\Psi}_{\tau_1}(\Phi)$;

* A regra $G_8 = suspeito(X) \rightarrow assassino(X)$ -($\exists Y\, suspeito\_assassino(Y)$ faz parte de $\Psi_{\tau_1}(\Phi)$, pois:
  $\mathrm{Lim}(G_8) = \exists Y\, suspeito\_assassino(Y)$
  $\mathrm{Hyp}(G_8) = suspeito(X) \rightarrow assassino(X)$
  Como consequência, $\mathrm{Lim}(G_8) \cap \mathrm{Th}_{\mathrm{LPD}}(\Phi \cup \mathrm{Hyp}(G_8)) = \emptyset$, e portanto, é verdade que:
  $\mathrm{Hyp}(G_8) \subseteq \Psi_{\tau_1}(\Phi)$ e $G_8 \in \overline{\Psi}_{\tau_1}(\Phi)$;

* A regra $G_3 = suspeito(X)$ -($inocente(X)$ apresenta uma peculiaridade muito interessante e que merece ser analisada com maior detalhe:
  $\mathrm{Lim}(G_3) = inocente(X)$
  $\mathrm{Hyp}(G_3) = suspeito(X)$

  Ocorre que o conjunto $\mathrm{Hyp}(G_3)$ pode ser reescrito como o conjunto $\{suspeito(dorothy)$, $suspeito(leland)$, $suspeito(rose)$ e $suspeito(scott)\}$. Além disso, por G_8, temos que $suspeito(dorothy) \rightarrow assassino(dorothy)$, $suspeito(leland) \rightarrow assassino(leland)$, $suspeito(rose) \rightarrow assassino(rose)$ e $suspeito(scott) \rightarrow assassino(scott)$ fazem parte de Φ, assim como a regra da unicidade do assassino de T. Dessa forma, qualquer que seja a instância G_3^i de G_3 que considerarmos primeiro, podemos provar a inocência de todos os outros elementos do domínio de suspeitos e, consequentemente, a exceção das outras instâncias, ou seja:

  $\mathrm{Lim}(G_3) \cap \mathrm{Th}_{\mathrm{LPD}}(\Phi \cup \mathrm{Hyp}(G_3)) = \{inocente(X) \mid suspeito(Y),\ X \neq Y,\ Y$ é único $\}$
  Seja $G_3^i \in \mathrm{Hyp}(G_3)$, uma instância da G_3, $\{G_3^i\} \subseteq \Psi_{\tau_1}(\Phi)$ e $G_3 \in \overline{\Psi}_{\tau_1}(\Phi)$;

A extensão calculada, é então:

$$E_1 = \mathrm{Th}_{\mathrm{LPD}}\Big(T \cup \big\{ \big(\forall x \, \big(\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_8)\big)\big)? \big\} \cup \Gamma\Big),$$

onde

$$\Gamma = \Big\{ \Big(suspeito(dorothy)\Big)?, \qquad \Big(suspeito(leland)\Big)?,$$
$$\Big(suspeito(rose)\Big)?, \qquad \Big(suspeito(scott)\Big)?\Big\}.$$

Observamos que, ao aplicar o operador sobre todo o conjunto gerado Φ, realmente encontramos um ponto fixo e, portanto, temos a extensão citada. Deste ponto em diante, apenas faremos o cálculo, omitindo essa informação ao fim de cada momento, pois a característica de ponto fixo ocorrerá em todos estes, tendo no conjunto Φ construído, a extensão. Inicialmente, ilustramos como é feito o cálculo, e ao fim de algumas sucessões ilustrativas, chegaremos ao cálculo de extensão que nos interessa, realizado no momento 9 com todas as informações obtidas.

Percebe-se ainda que a extensão reproduz possibilidades adversas dentro da mesma estrutura, através de um agrupamento de provas plausíveis marcadas pela modalidade ?, como ocorre neste nosso exemplo. A questão da funcionalidade destas provas plausíveis e sua mútua incompatibilidade é que cada uma dessas provas compõe um cenário junto aos conjuntos de hipóteses que não têm prova de suas exceções para nenhuma instância, e que ocorrem em todos os cenários. Enumeramos a seguir os cenários:

Seja $\Phi' = \Big\{T \cup \big\{ \big(\forall x \left(\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_8)\right)\big) \big\}\Big\}$:

$$S_1^1 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(dorothy)\}\Big);$$
$$S_1^2 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(leland)\}\Big);$$
$$S_1^3 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(rose)\}\Big);$$
$$S_1^4 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(scott)\}\Big).$$

Interpretamos este resultado como condições diferentes em que podemos enquadrar o problema enunciado, ou seja, somente podemos ter um assassino, mas as informações não nos dão nenhum direcionamento quanto à culpa de algum dos suspeitos já encontrados. Dessa forma, devemos ter um cenário em que Dorothy é a culpada, provada por

$suspeito(dorothy)$ e $suspeito(dorothy) \rightarrow assassino(dorothy)$, e em que todos os outros são inocentes. Esse quadro nos dará um conjunto específico de fórmulas que podem ser deduzidas a partir de suas suposições. Da mesma forma, ocorre para cada uma das outras pessoas que podem ser acusadas, e cada cenário só vai ter uma cláusula de assassino, ao menos até encontrarmos alguma evidência para um suspeito. Em breve, ilustraremos o que ocorre nessa ocasião.

Ressaltamos que cada cenário só apresenta um suspeito, pois sugere o que ocorre em termos clássicos se este suspeito for o assassino. Mesmo que a prova de assassino seja por outros meios, uma vez que ser suspeito é necessário à prova de assassino independente da quantidade de evidência reunida contra o elemento, consideramos apenas um culpado por vez e inocentamos às outras pessoas, sobrando apenas um suspeito (aquele que está sendo acusado). Isso é bastante interessante, pois a acusação deve ser feita sobre somente uma pessoa, e essa é escolhida quando o detetive descarta as demais opções, mesmo que por opinião própria, intuição ou qualquer outro tipo de "chute".

Dizemos ainda que cada um desses cenários prova todos os requisitos necessários para provar cada um destes elementos como assassino, exceto pelo requisito de serem considerados suspeitos, pois cada cenário só permite um suspeito (e consequentemente, uma única prova de assassino).

M2 O detetive verifica a cena do crime procurando por provas. Neste ambiente, encontramos quatro relevantes peças de evidência. São elas:

- Uma pegada marcada no chão, próximo ao local onde encontramos o cadáver. A pegada indica passagem de uma pessoa pesada.

- Fios de cabelo presos à roupa da vítima. Os fios encontrados são lisos.

- Fios de cabelo encontrados no chão. Esses fios de cabelo, embora diferentes dos encontrados na roupa da vítima, indicam uma segunda pessoa de cabelos lisos.

- Uma nota de ameaça à vítima é encontrada em seu bolso. Utilizando-se da habilidade de análise de caligrafia, é possível determinar que foi escrita por uma pessoa destra.

Informações obtidas na cena do crime são modeladas nas seguintes regras LPR e temos $\tau_2 = \langle T_2, G \rangle$, onde T_2 é T_1 acrescida em união com o conjunto de regras cujos elementos são os seguintes:

- O conjunto de provas agora é conhecido e consiste de 4 elementos.
- $destro(escrita)$;
- $liso(cabelo1)$;
- $liso(cabelo2)$;
- $\neg leve(pegada)$;

A extensão que obtemos a partir de τ_2 é calculada como segue:

- T_2 faz parte de $\Psi_{\tau_2}(\Phi)$.

- Quaisquer fórmulas que possam ser provadas a partir das fórmulas de $\Psi_{\tau_2}(\Phi)$, fazem parte de $\Psi_{\tau_2}(\Phi)$.

- Deixamos a cargo do leitor conferir que nada se altera em termos do conjunto de regras geradoras e instâncias de generalizações que passam a compor $\Psi_{\tau_2}(\Phi)$, uma vez que as regras adicionadas não provocam prova de nenhuma exceção dessas generalizações, nem ampliam os elementos suspeitos já encontrados.

A extensão calculada, é então:

$$E_2 = E_1 = \mathrm{Th}_{\mathrm{LPD}}\Big(T \cup \{\ (\forall x\ (\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_8)))?\ \} \cup \Gamma\Big),$$

onde

$$\Gamma = \Big\{\ \Big(suspeito(dorothy)\Big)?, \qquad \Big(suspeito(leland)\Big)?, \\ \Big(suspeito(rose)\Big)?, \qquad \Big(suspeito(scott)\Big)?\Big\}.$$

Da mesma forma, observamos os mesmos cenários $S_2^1 = S_1^1, \ldots, S_2^4 = S_1^4$.

Como pode ser notado, as informações adicionadas neste momento não fizeram diferença alguma com relação à ocorrência das generalizações nas teorias que estamos trabalhando. Os primeiros passos da estratégia citado têm intuito de encontrar todos os suspeitos, e algumas das regras sobre suspeitos são incompletas quando isso ocorre. Podemos utilizá-las sobre uma parte do conjunto de suspeitos, claro, mas isso pode nos levar a conclusões errôneas (pois supõe, de forma indutiva, que todos os suspeitos estão na lista). Partindo dessa idéia, vamos saltar alguns passos na estratégia até um ponto em que tenhamos todos os suspeitos e possamos começar a colher e verificar álibis e a inocência de alguns suspeitos.

M3 O detetive tem encontrado pessoalmente a todos os suspeitos. Além disso, ele aproveitou para realizar a pesquisa a que tem direito.
Com as novas informações obtidas, fazemos $\tau_3 = \langle T_3, G\rangle$, onde T_3 é T_2 acrescida em união com o conjunto de regras cujos elementos são os seguintes:

- $slim(carlile), man(carlile), \neg straight(carlile), \neg right(carlile)$;
- $slim(leland), man(leland), straight(leland), \neg right(leland)$;
- $slim(scott), man(scott), straight(scott), right(scott)$;
- $\neg slim(zoe), \neg man(zoe), \neg straight(zoe), \neg right(zoe)$;
- $\neg slim(chancey), man(chancey), straight(chancey), right(chancey)$;
- $slim(barnabas), man(barnabas), straight(barnabas), right(barnabas)$;
- $\neg slim(rose), \neg man(rose), \neg straight(rose), \neg right(rose)$;
- $\neg slim(susan), \neg man(susan), straight(susan), \neg right(susan)$;
- $pesquisa(carlile)$;

A seguir, calculamos a extensão obtida a partir de τ_3, como segue:

- T_3 faz parte de $\Psi_{\tau_3}(\Phi)$.
- Quaisquer fórmulas que possam ser provadas a partir das fórmu-

las de $\Psi_{\tau_3}(\Phi)$, fazem parte de $\Psi_{\tau_3}(\Phi)$.

- Sobre as generalizações:

  * As regras G_1, G_2 e G_4 apresentam $\text{Lim}(G_i) = \emptyset$. Essas regras se comportam da mesma forma que nos momentos anteriores, e por não terem exceções, continuarão desta forma, e portanto obedecem às condições para que $\text{Hyp}(G_i) \subseteq \Psi_{\tau_3}(\Phi)$ e $G_i \in \overline{\Psi}_{\tau_3}(\Phi)$. Passaremos a omitir essa explicação daqui em diante, atestando simplesmente que algumas regras seguem o mesmo comportamento que no momento imediatamente anterior, ou em algum momento especificado;

  * A regra G_5 apresenta $\text{Lim}(G_5) = \exists Y desfecho(Y)$. Como não é possível provar $desfecho(Y)$, para nenhum valor de Y, provamos que G_5 obedece às condições para que $\text{Hyp}(G_5) \subseteq \Psi_{\tau_3}(\Phi)$ e $G_5 \in \overline{\Psi}_{\tau_3}(\Phi)$;

  * A regra G_6 apresenta $\text{Lim}(G_6) = \exists Y provavel\_assassino(Y)$. Como não é possível provar $provavel\_assassino(Y)$, para nenhum valor de Y, provamos que G_6 obedece às condições para que $\text{Hyp}(G_6) \subseteq \Psi_{\tau_3}(\Phi)$ e $G_6 \in \overline{\Psi}_{\tau_3}(\Phi)$;

  * A regra G_7 apresenta $\text{Lim}(G_7) = \exists Y possivel\_assassino(Y)$. Como não é possível provar $possivel\_assassino(Y)$, para nenhum valor de Y, provamos que G_7 obedece às condições para que $\text{Hyp}(G_7) \subseteq \Psi_{\tau_3}(\Phi)$ e $G_7 \in \overline{\Psi}_{\tau_3}(\Phi)$;

  * A regra G_8 apresenta $\text{Lim}(G_8) = \exists Y suspeito\_assassino(Y)$. Como não é possível provar $suspeito\_assassino(Y)$, para nenhum valor de Y, provamos que G_8 obedece às condições para que $\text{Hyp}(G_8) \subseteq \Psi_{\tau_3}(\Phi)$ e $G_8 \in \overline{\Psi}_{\tau_3}(\Phi)$;

  * O que ocorre com a regra $G_3 = suspeito(X)$ *-(* $inocente(X)$, deve ser observado em maior detalhe:

    (1) Temos uma nova informação, ou seja, uma nova suposição a ser levada em conta na prova: $pesquisa(carlile)$.

    (2) Sabemos que $pesquisa(X) \rightarrow inocente(X)$;

(3) Por dedução, concluímos *inocente*(*carlile*);

(4) $\text{Lim}(G_3) = inocente(X)$;

(5) $\text{Hyp}(G_3) = suspeito(X)$;

(6) Similarmente ao que ocorreu no momento 1, por conta da regra de unicidade do assassino e a entrada das instâncias de G_8,

$\text{Lim}(G_3) \cap \text{Th}_{\text{LPD}}(\Phi \cup \text{Hyp}(G_3)) = \{inocente(carlile)\} \cup \{inocente(X) \mid suspeito(Y),\ X \neq Y,\ Y \neq carlile\, Y$ é único $\}$.

(7) Como a inocência de Carlile foi provada aquém de um suspeito específico ter sido descoberto assassino, a instância de G_3, *suspeito*(*carlile*) -(*inocente*(*carlile*) fica bloqueada e não faz mais parte deste extensão ou das próximas (o cálculo de *inocente*(*carlile*) é monotônico). Podemos incluir a $\Psi_{\tau_3}(\Phi)$, no entanto, as instâncias restantes, na forma *suspeito*(*dorothy*), *suspeito*(*leland*), *suspeito*(*scott*), *suspeito*(*barnabas*), *suspeito*(*susan*), *suspeito*(*zoe*), *suspeito*(*rose*) e *suspeito*(*chancey*).

A extensão calculada, é então:

$$E_3 = \text{Th}_{\text{LPD}}\Big(T \cup \big\{\ \big(\forall x\ \big(\bigwedge \text{Hyp}(G_1), \text{Hyp}(G_2), \text{Hyp}(G_4), \ldots, \text{Hyp}(G_8)\big)\big)?\ \big\} \cup \Gamma\Big),$$

onde

$$\begin{aligned}\Gamma = \Big\{\ &\Big(suspeito(dorothy)\Big)?, && \Big(suspeito(leland)\Big)?, \Big(suspeito(rose)\Big)?,\\ &\Big(suspeito(scott)\Big)?, && \Big(suspeito(barnabas)\Big)?, \Big(suspeito(chancey)\Big)?,\\ &\Big(suspeito(susan)\Big)?, && \Big(suspeito(zoe)\Big)?\Big\}.\end{aligned}$$

Notamos agora uma diferença bastante significativa. A nova informação adicionada *pesquisa*(*carlile*) foi suficiente para provar uma instância de $\text{Lim}(G_3)$, e por conta disso, retiramos da extensão que tínhamos antes a informação de que Carlile continua sendo suspeito, ou seja, a fórmula *suspeito*(*carlile*) deixa de ser provada pelo modelo, o que mostra adequadamente o funcionamento da regra $G_3 = suspeito(X)$

-($inocente(X)$, que modela a premissa "Todos são suspeitos até que se prove sua inocência". Da mesma forma, poderemos observar com outras generalizações a seguir, uma vez que nossa base de informações cresceu consideravelmente no momento 3. Ao encontrarmos todos os suspeitos, podemos começar a utilizar algumas regras que modelam provas por exclusão. Utilizamos essas regras para concluir algo sobre os suspeitos a partir da exclusão de todos os outros, e portanto, somente podemos fazer provas desse tipo quando todos os suspeitos são conhecidos.

Note que continuamos com vários cenários plausíveis, e que não existe nenhum cenário onde Carlile é suspeito. Pelo contrário, em todos eles, há uma prova de que Carlile é inocente obtida por uma simples aplicação do teorema da dedução. Os cenários que encontramos no momento 3 são, portanto:

Seja $\Phi' = \Big\{T \cup \{\ (\forall x\ (\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_8)))\ \}\Big\}$:

$$
\begin{aligned}
S_3^1 &= \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(dorothy)\}\Big);\\
S_3^2 &= \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(leland)\}\Big);\\
S_3^3 &= \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(rose)\}\Big);\\
S_3^4 &= \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(scott)\}\Big).\\
S_3^5 &= \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(barnabas)\}\Big);\\
S_3^6 &= \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(chancey)\}\Big);\\
S_3^7 &= \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(susan)\}\Big);\\
S_3^8 &= \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(zoe)\}\Big).
\end{aligned}
$$

Nossas informações sugerem que qualquer um pode ter cometido o crime, e não há nada que os diferencie em termos de culpa. Os oito cenários disponíveis nesse momento descrevem isso. Repare que as informações tiradas de instâncias de generalizações incompatíveis são postas em diferentes cenários, mas somente temos uma extensão na teoria.

M4 Essa é a hora de buscar e confirmar álibis. O detetive pergunta aos suspeitos por seus álibis. Não é necessário perguntar pelo álibi de Carlile, pois ele provavelmente dará um álibi falso, e não nos é interessantes gastar perguntas a ele ou aos NPC's averiguando algo de que já sabemos a resposta. Durante as entrevistas, os suspeitos Dorothy, Leland, Chancey, Barnabas, Rose e Susan nos dão álibis. Scott e Zoe já não querem responder perguntas e o detetive não consegue persuadí-los. Verificando os álibis, descobrimos junto aos NPC's que Leland, Chancey e Barnabas falam a verdade, os suspeitos Rose e Susan mentiram e não foi possível verificar o álibi de Dorothy. Assim, fazemos $\tau_4 = \langle T_4, G \rangle$, onde T_4 é T_3 acrescida em união com o conjunto de regras cujos elementos são os seguintes:

- $\neg alibi(carlile), \neg verdadeiro(carlile), \neg falso(carlile)$;
- $alibi(dorothy), \neg verdadeiro(dorothy), \neg falso(dorothy)$;
- $alibi(leland), verdadeiro(leland), \neg falso(leland)$;
- $\neg alibi(scott), \neg verdadeiro(scott), \neg falso(scott)$;
- $\neg alibi(zoe), \neg verdadeiro(zoe), \neg falso(zoe)$;
- $alibi(chancey), verdadeiro(chancey), \neg falso(chancey)$;
- $alibi(barnabas), verdadeiro(barnabas), \neg falso(barnabas)$;
- $alibi(rose), \neg verdadeiro(rose), falso(rose)$;
- $alibi(susan), \neg verdadeiro(susan), falso(susan)$;

A seguir, calculamos a extensão obtida a partir de τ_4, como segue:

- T_4 faz parte de $\Psi_{\tau_4}(\Phi)$.

- Quaisquer fórmulas que possam ser provadas a partir das fórmulas de $\Psi_{\tau_4}(\Phi)$, fazem parte de $\Psi_{\tau_4}(\Phi)$.

- Sobre as generalizações:

  * As regras G_1, G_2 e G_4, assim com as regras G_5, G_6 e G_7 apresentam o mesmo comportamento que no momento anterior.

  * G_8 apresenta instâncias bloqueadas. Vamos analisar: (1) As informações de álibi falso são suficientes para provar $\exists Y suspeito\_assassino(Y)$, exceção da regra. (2) Como a exceção dessa regra *é* uma cláusula existencial, *todas* as instâncias de G_8 são bloqueadas. (3) Como consequência, as instâncias

de G_3 não mais são suficientes para sugerir um assassino.

* Sobre a regra $G_3 = suspeito(X)$ -($inocente(X)$, as provas de $inocente(X)$, e consequentemente as instâncias que entram para a extensão, são ligeiramente diferentes. Segue:
(1) As novas informações de álibi verdadeiros, junto à regra $verdadeiro(X) \rightarrow inocente(X)$ sugerem mais inocentes.

(2) Podemos concluir $inocente(carlile)$, $inocente(barnabas)$, $inocente(chancey)$, $inocente(leland)$, dedutivamente em T;

(3) As informações de álibi falso são suficientes para provar $evidencia(susan)$ e $evidencia(rose)$, e $suspeito\_assassino(susan)$ e $suspeito\_assassino(rose)$; Essas regras, junto a instâncias de G_7 que entram na extensão, produzem possíveis provas de assassino e, consequentemente, mais que um cenário, similarmente ao que aconteceu anteriormente envolvendo as instâncias da regra G_3 com as instâncias de G_8. Dizemos:

$$\mathrm{Lim}(G_3) \cap \mathrm{Th}_{\mathrm{LPD}}(\Phi \cup \mathrm{Hyp}(G_3)) = \{inocente(carlile), inocente(barnabas), inocente(chancey), inocente(leland)\}$$
$$\cup \{inocente(X) \;\Big|\; suspeito(Y),\ X \neq Y,\ (Y \not\in \{carlile, barnabas, chancey, leland\}),\ Y \text{ é único } \}.$$

(4) Dessa vez, com provas de $suspeito\_assassino(susan)$ e $suspeito\_assassino(rose)$, somente duas instâncias de G_3 serão consideradas, uma vez que em cada caso todos os outros são inocentados, como nos momentos anteriores.

A extensão calculada, é então:

$$E_4 = \mathrm{Th}_{\mathrm{LPD}}\Big(T \cup \{\ (\forall x\ (\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_7)))?\ \} \cup \Gamma\Big),$$

onde

$$\Gamma = \Big\{ \Big(suspeito(susan)\Big)?, \qquad \Big(suspeito(rose)\Big)?,$$
$$\Big((\forall x\, \mathrm{Hyp}(G_8)) \wedge suspeito(dorothy)\Big)?,$$
$$\Big((\forall x\, \mathrm{Hyp}(G_8)) \wedge suspeito(scott)\Big)?,$$
$$\Big((\forall x\, \mathrm{Hyp}(G_8)) \wedge suspeito(zoe)\Big)? \Big\}.$$

Em vista dos álibis mentirosos das suseitas Susan e Rose, devemos focar nossas suspeitas nelas. Além disso, se precisássemos acusar alguém neste momento, somente com essas informações, sabemos pelo modelo que devemos escolher entre Susan e Rose. Isso pode ser notado pelo bloqueio das cláusulas de G_8 nos cenários em que elas são consideradas suspeitas, mostrando uma maior restrição nas possibilidades de afirmações factíveis. Os cenários gerados seguem abaixo:

Seja $\Phi' = \Big\{T \cup \{\ (\forall x\ (\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_7)))\ \}\Big\}$:

$$S_4^1 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(susan)\}\Big);$$
$$S_4^2 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(rose)\}\Big);$$
$$S_4^3 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{(\forall x\, \mathrm{Hyp}(G_8)) \wedge suspeito(dorothy)\}\Big);$$
$$S_4^4 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{(\forall x\, \mathrm{Hyp}(G_8)) \wedge suspeito(scott)\}\Big);$$
$$S_4^5 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{(\forall x\, \mathrm{Hyp}(G_8)) \wedge suspeito(susan)\}\Big).$$

Fechamos nossa busca a apenas duas pessoas, que são os suspeitos dos cenários mais restritos em termos de generalizações (e menos suscetíveis a reviravoltas), pois essas são as únicas que apresentaram evidências de culpa. O detetive deve, então focar seus esforços no sentido de buscar determinar a culpa de uma das duas em definitivo.

M5 O detetive agora começa a utilizar seus recursos de perguntas aos NPC's para descobrir combinações de prova e suspeito. Faz perguntas baseado nas pessoas que podem ter deixado as provas e em quem

são os mais suspeitos. Até agora, Rose e Susan mentiram, e procuraremos verificar junto aos NPC's se algumas das evidências físicas da cena do crime realmente pertencem a elas. Esses personagens somente respondem perguntas de álibis e indicam pessoas que testemunham ou fazem combinações com provas. Assim sendo, podemos utilizar todas as perguntas restantes buscando encontrar os donos dessas provas. Por questões de interesse nas generalizações, dividimos as questões de combinações de prova neste momento e nos dois seguintes. No momento 5, procuraremos provar a culpa de Susan ou Rose.
Podemos verificar, pelas suas características físicas e o que as provas evidenciam, que se Rose tiver produzido algumas prova, esta é a pegada. O detetive se dispõe a utilizar uma de suas perguntas com o sapateiro para verificar se a pegada pertence a Rose, mas não é o caso. Sobre susan, somente podemos verificar que a nota não foi escrita por ela, pois teria sido escrita por uma pessoa destra, mas susan é canhota. O detetive pergunta ao sapateiro e cabeleireiro se a pegada ou alguma amostra de cabelos pertencem a ela e descobre que não.
Assim sendo, fazemos $\tau_5 = \langle T_5, G \rangle$, onde T_5 é T_4 acrescida em união com o conjunto de regras cujos elementos são os seguintes:

- $nao\_match(pegada, rose)$;
- $nao\_match(pegada, susan)$;
- $nao\_match(cabelo1, susan)$;
- $nao\_match(cabelo2, susan)$;

A seguir, calculamos a extensão obtida a partir de τ_5, como segue:

- T_5 faz parte de $\Psi_{\tau_5}(\Phi)$.

- Quaisquer fórmulas que possam ser provadas a partir das fórmulas de $\Psi_{\tau_5}(\Phi)$, fazem parte de $\Psi_{\tau_5}(\Phi)$.

- Sobre as generalizações:

  * As regras G_1, G_2 e G_4, assim com as regras G_5, G_6 e G_7 apresentam o mesmo comportamento que no momento anterior.

  * G_8 deixa de apresentar instâncias bloqueadas, pois sua exceção não mais pode ser provada, e G_8 volta a ter o mesmo comportamento que nos momentos 3 e anteriores, onde sua generalização faz parte da extensão para qualquer valor a que seja aplicada.

$*$ Mais uma vez, ocorrem diferenças quanto à regra $G_3 = suspeito(X)$ -($inocente(X)$. Em maior detalhe:

(1) As novas informações de não combinação, junto à regra $\exists P((match(P, X) \wedge assassino(X))$ sugerem que Susan e Rose são, na verdade, inocentes.

(2) Podemos concluir agora, em T, que $inocente(susan)$ e $inocente(rose)$, além das instâncias $inocente(carlile)$, $inocente(barnabas)$, $inocente(chancey)$, $inocente(leland)$, que já tínhamos;

(3) Não há mais provas de $\exists Y possivel\_assassino(Y)$, pois é requisito ser suspeito, e as pessoas que têm evidência contra si foram inocentadas.

A extensão calculada, é então:

$$E_5 = \mathrm{Th}_{\mathrm{LPD}}\Big(T \cup \{\ (\forall x\ (\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_8)))?\ \} \cup \Gamma\Big),$$

onde

$$\Gamma = \Big\{\ \Big(suspeito(dorothy)\Big)?, \quad \Big(suspeito(scott)\Big)?, \Big(suspeito(zoe)\Big)?\Big\}.$$

Neste passo ocorre algo muito similar ao passo anterior, quanto ao que ocorre quando verificamos álibis *verdadeiros*. A única diferença foi um par de restrições a mais na lista de suspeitos. Essas restrições, claro restringem a quantidade de possibilidades e provas dentro de Φ. Estamos nos aproximando de algum resultado. Este passo foi colocado como intermediário para manter a sequência da estratégia, enquanto justificando o contexto do próximo momento.

Vale ressaltar que, quando procurarmos por uma resposta para o autor do assassinato, não será possível encontrar provas contra nenhum suspeito atual, ou seja, as novas informações foram suficientes para derrubar as alternativas de acusar Rose ou Susan, e todos os cenários considerados apresentam novamente o mesmo poder argumentativo contra o suspeito culpado. Isso ocorre porque, embora tenhamos provado $suspeito\_assassino(X)$ dedutivamente, utilizamos como premissa que elas eram suspeitas, ou seja, usamos $suspeito(susan)$ e $suspeito(rose)$, que foram concluídas de forma ampliativa. Os novos fatos permitem deduzir a inocência das duas e derrubam as suspeitas contra elas. Neste passo fica notório que a influência das regras não-monotônicas é bastante profunda na LPR, pois a consideramos uma verdade incondicional para efeitos de prova desde que sua exceção não seja provada, o que dá a cada afirmação desse tipo, no contexto da prova, o mesmo poder expressivo máximo de uma fórmula *hard* (enquanto não forem inclusas novas informações).

Continuamos com alguns cenários plausíveis. São eles:

Seja $\Phi' = \Big\{T \cup \{ \ (\forall x \ (\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_8))) \ \} \Big\}$:

$$S_5^1 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(dorothy)\}\Big);$$
$$S_5^2 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(scott)\}\Big);$$
$$S_5^3 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(zoe)\}\Big).$$

M6 Seguindo anotações sobre as informações já coletadas (ou realizando provas matemáticas sobre a informação modelada até T_5, podemos verificar algumas possibilidades particularmente interessantes para a investigação do caso:

(1) *pegada* é uma evidência que sugere uma pessoa pesada. Assim sendo, sabemos que o dono da pegada não está entre Carlile, Dorothy, Leland, Scott e Barnabas. Verificamos ainda, que a mesma não pertence a Susan ou Rose. Assim sendo, ficamos entre Zoe e Chancey como possíveis donos. Como Chancey é inocente, o detetive pergunta se a pegada pertence a Zoe. (2) *cabelo*2 é uma evidência que sugere uma pessoa de cabelos lisos. Assim sendo, sabemos que o dono da amostra não está entre Carlile, Dorothy, Rose e Zoe. Dessa forma, entre os suspeitos restantes, pode pertencer a Barnabas, Chancey, Leland, Scott, e sabemos que não pertence a Susan. Podemos observar que três dos possíveis donos são inocentes, ou seja, a amostra pertence a Scott, ou a algum inocente. Isso é bastante para valer o teste caso pertença a Scott.
Como resposta às perguntas e verificações, constatamos que a amostra de cabelo *cabelo*2 pertence a Scott, mas a amostra de pegada não pertence a Zoe. Desta forma, fazemos $\tau_6 = \langle T_6, G \rangle$, onde T_6 é T_5 acrescida

em união com o conjunto de regras cujos elementos são os seguintes:

- $nao\_match(pegada, zoe)$;
- $match(cabelo2, scott)$;

Ressaltamos que a primeira informação, como sugere o contexto em que ela é obtida, é suficiente para o modelo provar $match(pegada, chancey)$. A seguir, calculamos a extensão obtida a partir de τ_6, como segue:

- T_6 faz parte de $\Psi_{\tau_6}(\Phi)$.

- Quaisquer fórmulas que possam ser provadas a partir das fórmulas de $\Psi_{\tau_6}(\Phi)$, fazem parte de $\Psi_{\tau_6}(\Phi)$.

- Sobre as generalizações:

  * As regras G_1, G_2 e G_4, assim com as regras G_5, G_6 e G_7 apresentam o mesmo comportamento que no momento anterior.

  * G_8 apresenta instâncias bloqueadas, pois:
    (1) A informação de combinação em que a amostra cabelo2 pertence a Scott é suficiente para provar $\exists Y\, suspeito\_assassino(Y)$, exceção da regra. (2) Como a exceção dessa regra *é* uma cláusula existencial, *todas* as instâncias de G_8 são bloqueadas. (3) Como consequência, as instâncias de G_3 não mais são suficientes para sugerir um assassino.

  * Sobre a regra $G_3 = suspeito(X)$ *-($inocente(X)$*, as provas de $inocente(X)$, e consequentemente as instâncias que entram para a extensão, são ligeiramente diferentes. Segue:
    (1) As novas informações de não combinação, junto à regra $\exists P((match(P, X) \wedge assassino(X))$ sugerem que Zoe é, na verdade, inocente.

    (2) Podemos, de agora em diante, concluir $inocente(zoe)$, dedutivamente em T;

    (3) A informação de combinação em que a amostra cabelo2 pertence a Scott é suficiente para provar $evidencia(scott)$ e $suspeito\_assassino(scott)$. Essa afirmação, junto a instâncias de G_7 que entram na extensão, produzem uma possível prova

de assassino e, consequentemente, um cenário, muito similar ao que tivemos envolvendo Susan e Rose no momento 4. Dizemos:

$\mathrm{Lim}(G_3) \cap \mathrm{Th}_{\mathrm{LPD}}(\Phi \cup \mathrm{Hyp}(G_3)) = \{inocente(carlile), inocente(barnabas), inocente(zoe), inocente(susan), inocente(rose), inocente(chancey), inocente(leland)\}$
$\cup \{inocente(X) \mid suspeito(Y),\ X \neq Y,\ Y \in \{scott, dorothy\},\ Y$ é único $\}$.

(4) Dessa vez, com prova de *suspeito_assassino*(*scott*), ocorre que teremos apenas um cenário a ser considerado, e somente uma instância de G_3 será considerada. Isso não quer dizer ainda que tenhamos um assassino, mesmo sendo um único cenário, pois este cenário não permite provar um desfecho. Ele é construído por exigência da unicidade dos assassinos e é o único levado em conta neste momento, pois temos sete inocentes comprovados e simplesmente não é plausível acusar Dorothy, invés de Scott (por isso não aparece como um segundo cenário).

A extensão calculada, é então:

$$E_6 = \mathrm{Th}_{\mathrm{LPD}}\Big(T \cup \{\ (\forall x\ (\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_7)))?\ \} \cup \Gamma\Big),$$

onde

$$\Gamma = \Big\{\ \Big(suspeito(scott)\Big)?,\quad \Big(\forall x\, \mathrm{Hyp}(G_8)\ \wedge\ suspeito(dorothy)\Big)?\ \Big\}.$$

Assim sendo, os cenários gerados são:

Seja $\Phi' = \Big\{T \cup \{\ (\forall x\ (\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_7)))\ \}\Big\}$:

$$S_6^1 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(scott)\}\Big).$$

$$S_6^2 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{\forall x\, \mathrm{Hyp}(G_8)\ \wedge\ suspeito(dorothy)\}\Big).$$

Os cenários são tais que S_6^1 ilustra a possibilidade de Scott ser o assassino, e permitindo prova da exceção de G_8 de forma a excluir todas as suas instâncias e S_6^2 descreve o caso em que Dorothy é culpada, e como Scott é inocente, não temos prova de $suspeito\_assassino(scott)$ ou para qualquer outra pessoa, permitindo considerar as instâncias de G_8.

Assim sendo, caso busquemos prova de $assassino(X)$ nesse momento, encontramos que Scott tem uma evidência contra si e tem contra ele o argumento mais forte quanto a ser o assassino (o cenário com menos generalizações), similarmente ao que fizemos para Rose e Susan no momento 4, ou seja, temos prova de $suspeito\_assassino(scott)$, e ele seria nossa melhor, e a única opção lógica de acusação baseado nas informações que temos até agora.

M7 Dando continuidade às verificações de combinações, podemos observar, que:

(1) $cabelo1$ é uma amostra que somente pode pertencer a Barnabas ou Leland. Como o próprio modelo consegue provar em T_7, $cabelo1$ pertence a um inocente, o que é expresso em $match\_inocente(cabelo1)$. (2) $escrita$ é uma evidência que só pode pertencer a Barnabas ou Dorothy. Como Barnabas é inocente, verificaremos se pertence a Dorothy. Como resposta às perguntas e verificações, constatamos que a amostra de cabelo $escrita$ pertence a Dorothy, e dessa forma, fazemos $\tau_7 = \langle T_7, G \rangle$, onde T_7 é T_6 acrescida em união com o conjunto de regras cujos elementos são os seguintes:

- $match\_inocente(cabelo1)$;
- $match(escrita, dorothy)$;

A seguir, calculamos a extensão obtida a partir de τ_7, como segue:

- T_7 faz parte de $\Psi_{\tau_7}(\Phi)$.

- Quaisquer fórmulas que possam ser provadas a partir das fórmulas de $\Psi_{\tau_7}(\Phi)$, fazem parte de $\Psi_{\tau_7}(\Phi)$.

- Sobre as generalizações:

  * As regras $G_1, \ldots, G_8$ apresentam o mesmo comportamento que no momento anterior, ou extremamente similar.

No momento 7, observamos prova de $suspeito\_assassino(scott)$ e $suspeito\_assassino(doro$
As generalizações e instâncias que entram na extensão são as mesmas,

mas agora voltamos a ter dois cenários similares, em que as provas com relação a culpa de cada um são equivalentes entre os cenários.
A extensão calculada, é então:

$$E_7 = \mathrm{Th}_{\mathrm{LPD}}\Big(T \cup \{\ (\forall x\ (\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_7)))?\ \} \cup \Gamma\Big),$$

onde

$$\Gamma = \Big\{\ \Big(\mathit{suspeito}(\mathit{scott})\Big)?, \qquad \Big(\mathit{suspeito}(\mathit{dorothy})\Big)?\ \Big\}.$$

Assim sendo, os cenários gerados são:

Seja $\Phi' = \Big\{T \cup \{\ (\forall x\ (\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_7)))\ \}\Big\}$:

$$S_7^1 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{\mathit{suspeito}(\mathit{scott})\}\Big).$$
$$S_7^2 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{\mathit{suspeito}(\mathit{dorothy})\}\Big).$$

Os cenários são tais que S_7^1 ilustra a possibilidade de Scott ser o assassino, e permitindo prova da exceção de G_8 de forma a excluir todas as suas instâncias e S_7^2 descreve o caso em que Dorothy é culpada, e como Scott é inocente, não temos prova de $\mathit{suspeito\_assassino}(\mathit{scott})$ ou para qualquer outra pessoa, permitindo considerar as instâncias de G_8.

Nos resta buscar uma melhor resposta para decidir a quem acusar através dos testemunhos, e somente mais dois momentos a ilustrar.

M8 Sobre as últimas entrevistas por testemunhos, consideraremos dois momentos. O primeiro descreve a situação em que, quando procurando por testemunhos, conseguimos um testemunho contra Scott. Isso sugere $\tau_8 = \langle T_8, G \rangle$, onde T_8 é $T_7 \cup \{\mathit{testemunho}(\mathit{scott})\}$;

A seguir, calculamos a extensão obtida a partir de τ_8, como segue:

– T_8 faz parte de $\Psi_{\tau_8}(\Phi)$.

– Quaisquer fórmulas que possam ser provadas a partir das fórmulas de $\Psi_{\tau_8}(\Phi)$, fazem parte de $\Psi_{\tau_8}(\Phi)$.

– Sobre as generalizações:

* As regras G_1, G_2, G_3, G_4, G_5 e G_8 apresentam o mesmo comportamento que no momento anterior.

* A regra G_6 apresenta $\text{Lim}(G_6) = \exists Y provavel\_assassino(Y)$. Neste momento, com o testemunho contra Scott, passamos a provar $provavel\_assassino(scott)$. Como consequência, verificamos que todas as instâncias de G_6 serão bloqueadas, pois diferente de G_3, sua exceção é uma cláusula existencial;

* Observamos com regra G_7, o mesmo fenômeno que ocorre com a regra G_6. Uma vez que é possível provar $possivel\_assassino(scott)$ e sua exceção é uma cláusula existencial, bloqueamos todas as ocorrências de G_7;

A extensão calculada, é então:

$$E_8 = \text{Th}_{\text{LPD}}\Big(T \cup \{\, (\forall x\, (\bigwedge \text{Hyp}(G_1), \text{Hyp}(G_2), \text{Hyp}(G_4), \ldots, \text{Hyp}(G_5)))?\,\} \cup \Gamma\Big),$$

onde

$$\Gamma = \Big\{\, \Big(suspeito(scott)\Big)?,\quad \Big(\forall x\Big(\bigwedge\Big\{\text{Hyp}(G_6),\ \text{Hyp}(G_7),\ suspeito(dorothy)\Big\}\Big)\Big)?\,\Big\}.$$

Ao concluirmos as instâncias de predicados $possivel\_assassino(scott)$ e $provavel\_assassino(scott)$, todas as instâncias das regras G_6 e G_7 têm suas exceções provadas, no caso em que Scott realmente é o assassino (e se tornaria o único suspeito). Isso é bastante para que nenhuma dessas instâncias faça parte de tal cenário. Ocorre ainda um segundo cenário em que Scott é inocente, e portanto não pode ser o assassino. Neste caso, verificamos que o único suspeito que sobra é Dorothy, e temos o segundo cenário, da mesma forma, que permite provar $assassino(dorothy$, uma vez que não haja provas de $\exists Y(possivel\_assassino(Y))$ e $\exists Y(provavel\_assassino(Y))$, e as instâncias de G_6 e G_7 podem ser assumidas. Seguem abaixo os cenários:

Seja $\Phi' = \Big\{T \cup \{\ (\forall x\ (\bigwedge \mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4), \ldots, \mathrm{Hyp}(G_7)))\ \}\Big\}$:

$S_8^1 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(scott)\}\Big)$.

$S_8^2 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{\forall x\Big(\bigwedge\Big\{\mathrm{Hyp}(G_6),\ \mathrm{Hyp}(G_7),\ suspeito(dorothy)\Big\}\Big)\Big\}\Big)$.

O cenário que sugere Scott como culpado é mais expressivo nesse sentido, pois permite apontar a culpa de alguém que tem uma prova física, um testemunho contra, e de quem não conseguimos obter um álibi. Nota-se pela quantidade de generalizações a mais que são permitidas no segundo cenário, o que indica bastante uso a mais de intuição e suposições ampliativas do que no primeiro.

M9 Finalmente, vejamos o que acontece no último momento, quando continuamos procurando por testemunhos contra Scott e Dorothy. Na sequência de perguntas, não mais encontramos nenhuma outra pessoa que apresente testemunho contra Scott. Antes que isso aconteça, encontramos o primeiro e o segundo testemunhos contra Dorothy. Segue que as novas informações que compõem $\tau_9 = \langle T_8, G\rangle$, onde T_9 é $T_8 \cup \{match(escrita, dorothy)\}$;

A extensão obtida a partir de τ_9, é calculada, como segue:

- T_9 faz parte de $\Psi_{\tau_9}(\Phi)$.
- Quaisquer fórmulas que possam ser provadas a partir das fórmulas de $\Psi_{\tau_9}(\Phi)$, fazem parte de $\Psi_{\tau_9}(\Phi)$.
- Sobre as generalizações:
    * As regras $G_1, G_2, G_3, G_4, G_6, G_7$ e G_8 apresentam o mesmo comportamento que no momento anterior.
    * Observamos com a regra G_5, o mesmo fenômeno que ocorre com as regras G_6 e G_7 neste e no momento anterior. Uma vez que é possível provar $desfecho(dorothy)$ e sua exceção é uma cláusula existencial, bloqueamos todas as ocorrências de G_5, também;

A extensão calculada, é então:

$$E_8 = \mathrm{Th}_{\mathrm{LPD}}\Big(T \cup \{\ (\forall x\ (\bigwedge\{\mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4)\}))?\ \} \cup \Gamma\Big),$$

onde

$$\Gamma = \Big\{\ \Big(suspeito(dorothy)\Big)?,\ \Big\}.$$

Consequentemente, o nosso único cenário, é:
Seja $\Phi' = \Big\{T \cup \{\ (\forall x\ (\bigwedge\{\mathrm{Hyp}(G_1), \mathrm{Hyp}(G_2), \mathrm{Hyp}(G_4)\}))\ \}\Big\}$:

$$S_9^1 = \mathrm{Th}_{\mathrm{CL}}\Big(\Phi' \cup \{suspeito(dorothy)\}\Big).$$

Na conclusão do nosso processo de prova, conseguimos informações suficientes para provar a cláusula $desfecho(dorothy)$ em T_9, monotonicamente, e com isso, negar às instâncias da regra G_5, além de G_6, G_7 e G_8. Dessa forma, terminamos com somente um cenário, e uma a garantia de que não existem outras possibilidades de culpados. O detetive pode acusar a Dorothy, desta vez suportado por um resultado dedutivo, enquanto que em todos os outros momentos ele tinha, em uma única extensão, modelos para cada possível assassino, o que lhe permitiria analisar as possíveis soluções e tomar decisões sobre suas ações na sequência estratégica, como fizemos durante este processo. A natureza dedutiva desde cenário final pode ser verificado através do fato que as generalizações que fazem parte da extensão não possuem exceção alguma, e portanto não há como derrotarmos provas já realizadas sobre o cenário com a adição de novas informações.

Com o detalhamento do que ocorre a cada momento acima, ilustramos o funcionamento da *LPR* e o modelo de raciocínio não-monotônico retratado por esta. Através da adição de novas informações, colhidas entre momentos de análise do caso, pudemos derrubar suposições e possibilidades de culpados, bem como foi possível guiar alguns passos na busca por estas informações a fim de confirmá-las ou derrubá-las. Em vários momentos, tivemos pessoas que eram consideradas como principais suspeitos, mas em seguida buscamos confirmação ou negação destas hipóteses, e pudemos recorrigir nossa análise ao refazer o cálculo da extensão, corrigindo-a. Com isso, encerramos nossa ilustração do poder da lógica *LPR*. Ressaltamos que momento 9, a base monotônica T_9 é como segue:

- $slim(carlile), man(carlile), \neg straight(carlile), \neg right(carlile)$;
- $slim(dorothy), \neg man(dorothy), \neg straight(dorothy), right(dorothy)$;
- $slim(leland), man(leland), straight(leland), \neg right(leland)$;
- $slim(scott), man(scott), straight(scott), right(scott)$;
- $\neg slim(zoe), \neg man(zoe), \neg straight(zoe), \neg right(zoe)$;
- $\neg slim(chancey), man(chancey), straight(chancey), right(chancey)$;
- $slim(barnabas), man(barnabas), straight(barnabas), right(barnabas)$;
- $\neg slim(rose), \neg man(rose), \neg straight(rose), \neg right(rose)$;
- $\neg slim(susan), \neg man(susan), straight(susan), \neg right(susan)$;
- $\neg alibi(carlile), \neg verdadeiro(carlile), \neg falso(carlile)$;
- $alibi(dorothy), \neg verdadeiro(dorothy), \neg falso(dorothy)$;
- $alibi(leland), verdadeiro(leland), \neg falso(leland)$;
- $\neg alibi(scott), \neg verdadeiro(scott), \neg falso(scott)$;
- $\neg alibi(zoe), \neg verdadeiro(zoe), \neg falso(zoe)$;
- $alibi(chancey), verdadeiro(chancey), \neg falso(chancey)$;
- $alibi(barnabas), verdadeiro(barnabas), \neg falso(barnabas)$;
- $alibi(rose), \neg verdadeiro(rose), falso(rose)$;
- $alibi(susan), \neg verdadeiro(susan), falso(susan)$;
- $nao\_match(pegada, rose)$;
- $nao\_match(pegada, susan)$;
- $nao\_match(cabelo1, susan)$;
- $nao\_match(cabelo2, susan)$;
- $nao\_match(pegada, zoe)$;
- $match(cabelo2, scott)$;
- $match\_inocente(cabelo1)$;
- $pesquisa(carlile)$;
- $destro(escrita)$;
- $liso(cabelo1)$;
- $liso(cabelo2)$;
- $\neg leve(pegada)$;
- $testemunho(scott)$;
- $testemunho(dorothy)$;
- $duplotestemunho(dorothy)$;

Essa base é obtida através da junção de todos os incrementos feitos desde o momento 1, de forma que, ao invés de calcularmos cada momento, poderíamos simplesmente ter calculado o último, com esse conjunto de fórmulas. O motivo pelo qual o fazemos diferente, é que além de dividir dessa forma, em vários exemplos, e observarmos algumas diferentes possíveis situações em que o cálculo da extensão vai mostrar um resultado com particularidades interessantes, modelamos dessa forma o próprio raciocínio, discretizado em alguns poucos momentos de análise, envolvido na resolução de um caso deste jogo, ou instância de problema. A forma como nós, pessoas, sugerimos alguma resposta baseado em evidências parciais e depois nos retratamos diante de evidências mais fortes que apontem em outro sentido ou limitações em nossa teoria é muito similar ao que foi retratado nesta sequência construtiva.

Capítulo 5

Ferramentas de verificação do cálculo de extensões

No capítulo anterior, definimos predicados sobre os domínios de pessoas ou suspeitos e de provas, e especificamos, com isso, um modelo para o cálculo de extensões *LPR*. Esse cálculo é realizado de forma que verificamos, para cada regra de generalização, se a exceção desta pode ser provada a partir das regras que foram tomadas como verdade em conjunto com sua própria afirmação ou generalização. Embora analisemos uma regra por vez, fazemos o cálculo de forma que, na entrada de uma segunda regra, se a exceção da primeira for provada, a removemos do conjunto. É nosso interesse que as generalizações da extensão sejam compatíveis entre si e que se apresentem em conjunto maximal, e o cálculo proposto nos permite levar em conta a todas as generalizações simultaneamente.

Uma vez que nosso modelo tem um número bastante grande de regras, como instrumento de garantia dos cálculos realizados nos momentos que detalhamos no capítulo 3, propomos uma forma de testar se é possível provar as exceções de regras, bem como as instâncias de aplicações dos predicados definidos para maior detalhamento das extensões. Uma vez reunida toda a informação do caso e calculada sua extensão, podemos escolher um cenário em que trabalhar e fazer provas dedutivas sobre seu conteúdo. Assim podemos fazer no mecanismo sugerido aqui sob a forma de instruções de *PROLOG*.

Além dos programas que aqui mencionamos, chamamos novamente atenção para a ferramenta *Dr. Watson*, previamente citada, que permite anotações de forma organizada em uma tabela, e realiza algumas deduções sobre o conjunto de informações aí inseridas. Podemos utilizá-la para inserir as informações citadas em cada momento e verificar que as pessoas sugeridas como suspeitos no cálculo de extensões confere com aquelas que ainda não puderam ser inocentadas. Diferente deste meio de verificação, os programas que apresentamos permitem apontar algum suspeito que tenha mais chances aparentes de ser o assassino do que os outros, ou ao menos apontar o grupo daqueles que parecem mais comprometidos. *Dr. Whatson* somente aponta alguém quando é definitivo, e somente uma pessoa. No formalismo proposto pela *LPR* sob a forma de extensões, nos interessa averiguar os cenários plausíveis que permitam respostas à pergunta de quem é o autor do crime. Os programas a seguir permitem tanto trabalhar sobre um único cenário, como observar quais suspeitos têm maior nível de evidência contra si. Além disso, através da interpretação dos resultados que podem ser provados por estes programas, é também possível apontar cada pessoa não inocentada e seus níveis de comprometimento.

O modelo que propomos para o jogo *Sleuth* define um conjunto de predicados que retratam diferentes níveis de comprometimento. Temos regras que sugerem acusação a qualquer suspeito, em qualquer nível de comprometimento, desde que não haja nenhum outro que já esteja mais comprometido. Podemos observar que as exceções de cada uma dessas regras sugere uma hierarquia de culpa, e isto é amplamente utilizado em função de escolher uma pessoa a quem acusar. Além disso, se propusermos o conjunto de especificações do modelo, estratégia de investigação e sistema de provas do programa como um único sistema de apoio à decisão, essa hierarquia pode ser utilizada para produzir uma função de utilidade a ser aplicada sobre os cenários gerados com o intuito de escolher um suspeito a quem acusar em casos de informações inconclusivas. Esse princípio se assemelha bastante com o conceito básico por trás de teoria dos jogos, onde devemos observar a todos os possíveis cenários, e então dar uma nota a cada um destes e, finalmente, escolher a ação que nos leva ao melhor resultado. Da mesma forma seria em nosso sistema, pois a extensão deriva todos os cenários possíveis, a função daria uma nota, e escolhermos acusar àquele suspeito previsto como culpado no cenário que recebeu a melhor atribuição de nota. No nosso modelo, tal função de utilidade pode ser tão simples quanto atribuir valores 1, 10, 100, 1000 e 10000, respectivamente, para provas de suspeito(X), suspeito_assassino(X), possível_assassino(X), provável_assassino(X) e assassino(X), pois as diferenças teriam sempre fatores de 10 e, como podemos verificar nos cálculos do capítulo anterior, os cenários apontam o maior nível de culpa que pode ser atribuído a cada suspeito dentro daquele contexto. Assim sendo, descartamos a necessidade de avaliar os cenários para determinar uma função deste tipo, visto que ela pode ser tirada diretamente da hierarquia de culpa e seria um resultado pouco interessante. Mais animador que criar uma função de utilidade desse tipo é observar o fenômeno recém descrito, de que a hierarquia de culpa sugere a função e descarta, assim, sua necessidade.

Em seguida, mencionamos algumas dificuldade e forma de resolver empregadas na construção de um provador *PROLOG* para Sleuth baseado em nosso modelo, e apresentamos os códigos utilizados. Algumas técnicas bastante curiosas foram utilizadas, cabendo a explicação do motivo e forma de utilização de dois programas funcionando em paralelo, invés de um só, como temos apenas um modelo. De forma similar às especificações da extensão *LPR* e condições que regem a construção de cada cenário por ponto-fixo, sugere o que fazemos com estes programas.

5.1 Programas PROLOG

Para especificar nosso modelo em instruções prolog, algumas adaptações tiveram de ser feitas. Em princípio, por limitações e restrições da linguagem, como a não existência de operadores de equivalência, negação e quantificador existencial / universal explícitos, e depois por necessidade de contornar e controlar a forma como o prolog segue suas provas por *backtracking*. Para codificar e realizar testes nos códigos que seguem, utilizamos as implementações livres *gprolog* ('http://www.gprolog.org/'), e *SWI Prolog* ('http://www.swi-prolog.org/'), principalmente a primeira.

Em seguida, uma prévia dos problemas encontrados e o que utilizamos para contornar a estes e às exigências da linguagem.

- Falta do quantificador universal $\forall$:
  O quantificador não existe para uso explícito, ocorrendo implicitamente em qualquer regra que envolva variáveis ocorrendo tanto na cabeça quanto cauda das mesmas. A forma que encontramos de modelar regras que necessitassem deste operador, foi a enumeração das instâncias de predicados sobre todos os elementos do domínio apropriado, quando necessário.

- Falta do quantificador existencial $\exists$:

Esse quantificador não é disponibilizado, mas pode ser simulado em alguns casos através de uma condição de prova de predicados sobre uma nova variável. Por exemplo, nas nossas generalizações acusação por níveis de culpa, as exceções são uma ocorrência dos predicados das exceções sobre qualquer valor, o que é alcançado pelo prolog utilizando-se simplesmente outra variável. Outra alternativa, quando o existencial deveria aparecer na cabeça das regras, é enumerar todas as possibilidades.

- Falta da negação $\neg$:
  Em nosso modelo, fazemos amplo uso da negação clássica. A implementação deste operador em prolog tem sido um desafio, pois no contexto da relação de prova sugerida pela linguagem, a negação clássica é incompleta. Expressamos predicados que são trabalhados junto com instâncias de sua negação sob a forma de dois predicados, opostos, e utilizamos afirmações das instâncias apropriadas a cada caso. Isso não é o bastante para resolver completamente o problema, pois ainda precisamos utilizar de equivalências para conseguirmos tirar conclusões sobre essas cláusulas, o que nos leva ao próximo item.

- Falta da equivalência $\leftrightarrow$:
  Como não há operador relacional de equivalência, ou não equivalência, mantemos um problema quanto à representação de cláusulas positivas e nagativas de um mesmo predicado. A forma intuitiva de contornar a falta deste operador, é utilizar de duas implicações, mas o operador de consequência que o prolog emprega é operado ligeiramente da equivalência clássica e pode gerar programas não totais (com ciclos) por contra disso.

- Falta dos operadores de igualdade $=$ ou diferença $\neq$:
  Esses operadores são importantes para especificar regras de unicidades do assassino, combinações de cada prova e combinação com o assassino. Contornamos este problemas através de algumas definições por casos e a criação dos predicados $diferente(X,Y)$ para pessoas e $diferente_P(P,Q)$ para provas. Estes predicados são, também, especificados por enumeração.

- Regras do tipo “Existe um único”:
  Essas regras são expressas no nosso modelo com uso de quantificadores e a igualdade, por exemplo, nas regras “$(assassino(X) \wedge assassino(Y)) \rightarrow (X = Y)$” e “$\exists X(assassino(X))$ -(” para dizer que só há um assassino. Como alternativa, especificamos essas regras de forma compacta na seguinte:

  – $\exists Y \forall X(((X \neq Y) \rightarrow \neg assassino(X)) \wedge assassino(Y))$

  A regra acima é muito similar à que prova em nosso modelo que no caso de provarmos todas menos uma pessoas inocentes, temos $desfecho(X)$, onde X é essa única pessoa não inocente. Nota-se a utilização de quantificadores e igualdade, e contornamos esse problema através da enumeração para o existencial de assassino, e uma regra que permite provar inocência de todos os outros suspeitos caso algum assassino tenha sido descoberto ou seja atribuído verdade entre as regras do programa (Para o caso de análise dos cenários). Para representá-la, precisamos, assim como ocorre para a equivalência e igualdade, dividir em duas partes que representam cada um dos lados das provas, mas o operador de consequência do prolog gera um ciclo para esse par de regras.

- Problemas com predicados de dois argumentos não instanciados:
  Quando utilizamos predicados com dois parâmetros e pesquisamos pela sua verificação, o prolog tende a buscar valores que tornam a afirmação verdadeira, mesmo quando isso não é o que queremos, ou seja, definições não instanciadas nas caudas de regras, quando acessadas pelo backtracking, tendem a levar a prova em uma direção divergente da esperada e prejudicar às provas. Mais uma vez, resolvemos esses casos com substituições das instâncias do predicado e enumeração de um dos argumentos (tornando-o um predicado unário).

- Ausência de instâncias dos predicados nos momentos iniciais:
  Quando começamos a busca por informações, muitos destes predicados não estão instanciados. Para resolver este problema, definimos novas constantes nulas (nenhumsuspeito, nenhumaprova, nulo), de acordo com a necessidade, e utilizamos instâncias sobre esses elementos.

Sobre os ciclos gerados pelas regras complementares que remontam uma equivalência, devemos explicar mais. Em primeiro lugar, é necessário explicar o motivo pelo qual a utilização da regra de corte (!) não resolve estes ciclos. Uma vez que temos várias formas de chegar aos resultados e trabalhamos com regras para um provador de teoremas, ao reunir várias regras, se qualquer uma destas fizer uso do corte, as outras regras não visitadas são ignoradas na sequência de prova, o que faz com que nossas provas para alguns predicados deixem de ser totais. Como normalmente nossas consultas são em busca de possíveis valores em que um predicado pode ser aplicado, esses cortes se mostraram prejudiciais. Dessa forma, além de lidar com as limitações da linguagem, tivemos que encontrar outra forma de impedir os loops e controlar o fluxo das chamadas a predicados dentro das provas.

Nesse contexto, após muitas tentativas de resolver o problema dos laços de diversas formas, utilizamos uma tática para controlar as chamadas de predicados em regras que causavam ciclos. Visto da ausência de conceitos procedurais do prolog, e a ausência do conceito de iteração e contagem em seus programas (pelo menos no que se refere à não utilização de inteiros), e inspirado em resultados de ciência da computação quanto à utilização da operação primitiva de composição de programas, resolvemos dividir os pares de cláusulas problemáticas em dois programas. Basicamente, separamos as cláusulas que provam um certo predicados, daquelas que provam sua negação, ou seja, provas de $inocente(X)$ e $assassino(X)$, de $nao\_match(P, X) e match(P, X)$, e de $match\_inocente(P)$ e $match\_assassino(P)$. Ressaltamos que os primeiros predicados em cada par destes ocorre para todos os elementos do domínio, menos algum elemento em específico, enquanto os outros predicados ocorrem, cada, para somente um valor ou par de valores de suas variáveis. Assim sendo, especificamos os programas, e a seguir comentamos sobre a operação que lhes garante equivalência ao programa único que desejamos ter.

Escrevemos abaixo o códigos dos dois programas que criamos, chamados “Sleuth Inocente” e “Sleuth Culpado”. Os códigos estão prontos para serem copiados para um arquivo de extensão .pl e operados em qualquer implementação de prolog compatível em sintaxe com as implementações gprolog e SWI Prolog, e para qualquer instância de casos de Sleuth. Especificamos mais à frente o código das informações que coletamos e que descrevem o caso do nosso exemplo utilizado no capítulo anterior. Entradas e saídas de provas estão citadas nos comentários dos programas.

Programa “Sleuth Inocente”:

```
/* “Entradas” são as cláusulas match(P,X), match_assassino(P) e assassino(X) */

/* “Saídas” são provas de nao_match(P,X), inocente(X) e match_inocente(P) */

/* ==== Instâncias de entrada: ==== */

match_assassino(nenhumaprova).
assassino(nulo).
match(nenhumaprova, nenhumsuspeito).

/* ==== Fim Instâncias de entrada: ==== */

/* ================================
««« PROVA NAO_MATCH(P,X): »»»
================================ */

nao_match(P,X) :- nota(P),destro(P),left(X).
nao_match(P,X) :- nota(P),canhoto(P),right(X).
nao_match(P,X) :- hair(P),liso(P),curly(X).
nao_match(P,X) :- hair(P),crespo(P),straight(X).
nao_match(P,X) :- shoe(P),leve(P),heavy(X).
nao_match(P,X) :- shoe(P),pesado(P),slim(X).
nao_match(P,X) :- sexo(P),mulher(P),man(X).
nao_match(P,X) :- sexo(P),homem(P),woman(X).

nao_match(P,Y1) :- match(P,Y2) , diferente(Y1,Y2).
nao_match(P1,X) :- match(P2,X) , diferente_p(P1,P2).

/* ================================
««« PROVA MATCH_INOCENTE(P): »»»
================================ */

match_inocente(escrita) :- match(escrita,X), inocente(X).
match_inocente(cabelo1) :- match(cabelo1,X), inocente(X).
match_inocente(cabelo2) :- match(cabelo2,X), inocente(X).
match_inocente(pegada) :- match(pegada,X), inocente(X).
match_inocente(P) :- match_assassino(Q), diferente_p(P,Q).
```

```
/* ================================
««« PROVA INOCENTE(X): »»»
================================ */

inocente(X) :- verdadeiro(X).
inocente(X) :- pesquisa(X).
inocente(X) :- morto(X).
inocente(X) :- nao_match(escrita,X), nao_match(cabelo1,X), nao_match(cabelo2,X),
nao_match(pegada,X).
inocente(X) :- assassino(Y), diferente(X,Y).
```

Este programa apresenta somente as cláusulas de prova para as cláusulas que mais ocorrem, e utiliza inclusive regras a partir de ocorrências das suas negações, garantindo que só haja uma ocorrência de cada cláusula de entrada, ou, alternativamente, produzindo um absurdo (levando a uma extensão em que todos são inocentes, mas existem dois assassinos, e, consequentemente, dois inocentes assassinos).

A seguir, o código do segundo programa.

Programa Culpado:

/* "Entrada"são as cláusulas nao_match(P,X), inocente(X) e match_inocente(P) */

/* Provas de match(P,X), match_assassino(P) e assassino(X) */

/* ==== Instâncias de entrada: ==== */

inocente(nenhumsuspeito).
match_inocente(nenhumsuspeito).
nao_match(nenhumaprova, nenhumsuspeito).

/* ==== Fim Instâncias de entrada: ==== */

/* ============================
««« PROVAS DE MATCH(P,X) »»»
============================ */
match(P, carlile) :- nao_match(P, dorothy), nao_match(P, leland), nao_match(P, scott), nao_match(P, barnabas), nao_match(P, zoe), nao_match(P, susan), nao_match(P, rose), nao_match(P, chancey).

match(P, dorothy) :- nao_match(P, carlile), nao_match(P, leland), nao_match(P, scott), nao_match(P, barnabas), nao_match(P, zoe), nao_match(P, susan), nao_match(P, rose), nao_match(P, chancey).

match(P, leland) :- nao_match(P, carlile), nao_match(P, dorothy), nao_match(P, scott), nao_match(P, barnabas), nao_match(P, zoe), nao_match(P, susan), nao_match(P, rose), nao_match(P, chancey).

match(P, scott) :- nao_match(P, carlile), nao_match(P, dorothy), nao_match(P, leland), nao_match(P, barnabas), nao_match(P, zoe), nao_match(P, susan), nao_match(P, rose), nao_match(P, chancey).

match(P, barnabas) :- nao_match(P, carlile), nao_match(P, dorothy), nao_match(P, leland), nao_match(P, scott), nao_match(P, zoe), nao_match(P, susan), nao_match(P, rose), nao_match(P, chancey).

match(P, zoe) :- nao_match(P, carlile), nao_match(P, dorothy), nao_match(P, leland), nao_match(P, scott), nao_match(P, barnabas), nao_match(P, susan), nao_match(P, rose), nao_match(P, chancey).

match(P, susan) :- nao_match(P, carlile), nao_match(P, dorothy), nao_match(P, leland), nao_match(P, scott), nao_match(P, barnabas), nao_match(P, zoe), nao_match(P, rose), nao_match(P, chancey).

match(P, rose) :- nao_match(P, carlile), nao_match(P, dorothy), nao_match(P, leland), nao_match(P, scott), nao_match(P, barnabas), nao_match(P, zoe), nao_match(P, susan), nao_match(P, chancey).

match(P, chancey) :- nao_match(P, carlile), nao_match(P, dorothy), nao_match(P, leland), nao_match(P, scott), nao_match(P, barnabas), nao_match(P, zoe), nao_match(P, susan), nao_match(P, rose).

/* ==============================
««« PROVAS DE MATCH_ASSASSINO(P) »»»
============================== */
match_assassino(escrita) :- match_inocente(cabelo1), match_inocente(cabelo2), match_inocente(pegada).

match_assassino(cabelo1) :- match_inocente(escrita), match_inocente(cabelo2), match_inocente(pegada).

match_assassino(cabelo2) :- match_inocente(escrita), match_inocente(cabelo1), match_inocente(pegada).

match_assassino(pegada) :- match_inocente(escrita), match_inocente(cabelo1), match_inocente(cabelo2).

/* ==============================
««« PROVAS DAS EVIDÊNCIAS E NÍVEIS DE CULPA »»»
============================== */

prova(X) :- match(escrita,X).
prova(X) :- match(cabelo1,X).
prova(X) :- match(cabelo2,X).
prova(X) :- match(pegada,X).

quaseconclusao(X) :- prova(X),testemunho(X), diferente(X,nenhumsuspeito).

evidencia(X) :- falso(X), diferente(X,nenhumsuspeito).
evidencia(X) :- testemunho(X), diferente(X,nenhumsuspeito).
evidencia(X) :- prova(X), diferente(X,nenhumsuspeito).

suspeito(carlile) :- \+ inocente(carlile).
suspeito(dorothy) :- \+ inocente(dorothy).
suspeito(barnabas) :- \+ inocente(barnabas).
suspeito(leland) :- \+ inocente(leland).
suspeito(scott) :- \+ inocente(scott).
suspeito(zoe) :- \+ inocente(zoe).
suspeito(susan) :- \+ inocente(susan).
suspeito(rose) :- \+ inocente(rose).
suspeito(chancey) :- \+ inocente(chancey).

desfecho(X) :- falso(X),testemunho(X), diferente(X,nenhumsuspeito).
desfecho(X) :- duplotestemunho(X), diferente(X,nenhumsuspeito).
desfecho(X) :- falso(X),prova(X), diferente(X,nenhumsuspeito).

desfecho(X) :- match_assassino(P), match(P,X), !, nl, write('ASSASSINO! (Unico nao inocente com combinacao de prova):').

desfecho(carlile) :- inocente(dorothy), inocente(leland), inocente(scott), inocente(barnabas), inocente(zoe), inocente(susan), inocente(rose), inocente(chancey), !.

desfecho(dorothy) :- inocente(carlile), inocente(leland), inocente(scott), inocente(barnabas), inocente(zoe), inocente(susan), inocente(rose), inocente(chancey), !.

desfecho(leland) :- inocente(carlile), inocente(dorothy), inocente(scott), inocente(barnabas), inocente(zoe), inocente(susan), inocente(rose), inocente(chancey), !.

desfecho(scott) :- inocente(carlile), inocente(dorothy), inocente(leland), inocente(barnabas), inocente(zoe), inocente(susan), inocente(rose), inocente(chancey), !.

desfecho(barnabas) :- inocente(carlile), inocente(dorothy), inocente(leland), inocente(scott), inocente(zoe), inocente(susan), inocente(rose), inocente(chancey), !.

desfecho(zoe) :- inocente(carlile), inocente(dorothy), inocente(leland), inocente(scott), inocente(barnabas), inocente(susan), inocente(rose), inocente(chancey), !.

desfecho(susan) :- inocente(carlile), inocente(dorothy), inocente(leland), inocente(scott), inocente(barnabas), inocente(zoe), inocente(rose), inocente(chancey), !.

desfecho(rose) :- inocente(carlile), inocente(dorothy), inocente(leland), inocente(scott), inocente(barnabas), inocente(zoe), inocente(susan), inocente(chancey), !.

desfecho(chancey) :- inocente(carlile), inocente(dorothy), inocente(leland), inocente(scott), inocente(barnabas), inocente(zoe), inocente(susan), inocente(rose), !.
suspeito_assassino(X) :- suspeito(X),evidencia(X).

possivel_assassino(X) :- suspeito(X),quaseconclusao(X).

provavel_assassino(X) :- possivel_assassino(X), \+ alibi(X).

/* ==============================
««« PROVAS DE ASSASSINO(X) »»»
============================== */
assassino(X) :- suspeito(X), desfecho(X), !, nl, write('ASSASSINO! (Desfecho):').

assassino(X) :- provavel_assassino(X), \+ desfecho(Y), nl, write('Provavel Assassino (prova fisica, testemunho, sem alibi):').

assassino(X) :- possivel_assassino(X), \+ provavel_assassino(Y), nl, write('Possivel Assassino (prova fisica, testemunho):').

assassino(X) :- suspeito_assassino(X), \+ possivel_assassino(Y), nl, write('Suspeito Assassino (alguma evidencia comprovada):').

assassino(X) :- suspeito(X), \+ suspeito_assassino(Y), nl, write('Suspeito (inocência não pode ser provada):').

O segundo programa apresenta provas das cláusulas opostas às do primeiro, e de ocorrências dos predicados que são utilizados como ferramentas e provas parciais em direção a essas. Ressaltamos a complementariedade dos programas, e que as entradas de um são produzidas pelo outro e vice-versa. Ainda que sejam opostos, temos regras que são introduzidas por modelo por meio das investigações, e estas informações são comuns aos dois programas. Quando inseridas nos arquivos de ambos os programas, permitirão realizarmos as operações necessárias para que estes programas funcionem em conjunto como desejado. A seguir, modelamos as informações comuns. Em sua maioria, são instâncias de ocorrências de predicados que somente podem ser afirmadas quando nos é dito por algum personagem ou constatado na coleta de provas e entrevistas, não tendo regras para prová-las. As exceções a isso são postas no final, para enfatizar a diferença e facilitar a junção do código em cada programa, uma vez que o prolog pede que as regras sejam agrupadas pelo conteúdo de suas cabeças.

Segue o código comum aos dois programas.

Parte comum e definições de predicados necessários à operação com o prolog:

```
/* ==============================
««« INFORMAÇÕES GERAIS: »»»
============================== */

alibi(barnabas).
alibi(chancey).
alibi(dorothy).
alibi(leland).
alibi(rose).
alibi(susan).
canhoto(nenhumaprova).
crespo(nenhumaprova).
curly(carlile).
curly(dorothy).
curly(rose).
curly(zoe).
destro(escrita).
diferente_p(cabelo1, cabelo2).
diferente_p(cabelo1, escrita).
diferente_p(cabelo1, pegada).
diferente_p(cabelo2, cabelo1).
```

```
diferente_p(cabelo2, escrita).
diferente_p(cabelo2, pegada).
diferente_p(escrita, cabelo1).
diferente_p(escrita, cabelo2).
diferente_p(escrita, pegada).
diferente_p(pegada, cabelo1).
diferente_p(pegada, cabelo2).
diferente_p(pegada, escrita).
diferente(barnabas, carlile).
diferente(barnabas, chancey).
diferente(barnabas, dorothy).
diferente(barnabas, leland).
diferente(barnabas, rose).
diferente(barnabas, scott).
diferente(barnabas, susan).
diferente(barnabas, zoe).
diferente(carlile, barnabas).
diferente(carlile, chancey).
diferente(carlile, dorothy).
diferente(carlile, leland).
diferente(carlile, rose).
diferente(carlile, scott).
diferente(carlile, susan).
diferente(carlile, zoe).
diferente(chancey, barnabas).
diferente(chancey, carlile).
diferente(chancey, dorothy).
diferente(chancey, leland).
diferente(chancey, rose).
diferente(chancey, scott).
diferente(chancey, susan).
diferente(chancey, zoe).
diferente(dorothy, barnabas).
diferente(dorothy, carlile).
diferente(dorothy, chancey).
diferente(dorothy, leland).
diferente(dorothy, rose).
diferente(dorothy, scott).
diferente(dorothy, susan).
diferente(dorothy, zoe).
diferente(leland, barnabas).
```

```
diferente(leland, carlile).
diferente(leland, chancey).
diferente(leland, dorothy).
diferente(leland, rose).
diferente(leland, scott).
diferente(leland, susan).
diferente(leland, zoe).
diferente(rose, barnabas).
diferente(rose, carlile).
diferente(rose, chancey).
diferente(rose, dorothy).
diferente(rose, leland).
diferente(rose, scott).
diferente(rose, susan).
diferente(rose, zoe).
diferente(scott, barnabas).
diferente(scott, carlile).
diferente(scott, chancey).
diferente(scott, dorothy).
diferente(scott, leland).
diferente(scott, rose).
diferente(scott, susan).
diferente(scott, zoe).
diferente(susan, barnabas).
diferente(susan, carlile).
diferente(susan, chancey).
diferente(susan, dorothy).
diferente(susan, leland).
diferente(susan, rose).
diferente(susan, scott).
diferente(susan, zoe).
diferente(zoe, barnabas).
diferente(zoe, carlile).
diferente(zoe, chancey).
diferente(zoe, dorothy).
diferente(zoe, leland).
diferente(zoe, rose).
diferente(zoe, scott).
diferente(zoe, susan).
diferente(barnabas,nenhumsuspeito).
diferente(zoe,nenhumsuspeito).
```

```
diferente(dorothy,nenhumsuspeito).
diferente(susan,nenhumsuspeito).
diferente(rose,nenhumsuspeito).
diferente(leland,nenhumsuspeito).
diferente(chancey,nenhumsuspeito).
diferente(carlile,nenhumsuspeito).
diferente(scott,nenhumsuspeito).
falso(rose).
falso(susan).
hair(cabelo1).
hair(cabelo2).
heavy(chancey).
heavy(rose).
heavy(susan).
heavy(zoe).
homem(nenhumaprova).
left(carlile).
left(leland).
left(rose).
left(susan).
left(zoe).
leve(nenhumaprova).
liso(cabelo1).
liso(cabelo2).
man(barnabas).
man(carlile).
man(chancey).
man(leland).
man(scott).
morto(nenhumsuspeito).
mulher(nenhumaprova).
nota(escrita).
pesado(pegada).
pesquisa(carlile).
right(barnabas).
right(chancey).
right(dorothy).
right(scott).
sexo(nenhumaprova).
shoe(nenhumaprova).
shoe(pegada).
```

```
slim(barnabas).
slim(carlile).
slim(dorothy).
slim(leland).
slim(scott).
straight(barnabas).
straight(chancey).
straight(leland).
straight(scott).
straight(susan).
verdadeiro(barnabas).
verdadeiro(chancey).
verdadeiro(leland).
woman(dorothy).
woman(rose).
woman(susan).
woman(zoe).

match(escrita, dorothy).
match(cabelo2, scott).

testemunho(scott).
testemunho(dorothy).

duplotestemunho(dorothy).

nao_match(pegada,zoe).

nao_match(pegada, rose).
nao_match(pegada, susan).
nao_match(cabelo1, susan).
nao_match(cabelo2, susan).

match_inocente(cabelo1).
```

5.2 Operação dos programas

Agora que temos definidos os códigos que representam as regras do modelo *LPR* sugerido no capítulo 3, nos falta apenas estabelecer a exata relação dos dois programas em termos semânticos para que possam produzir produzir resultados de acordo com o que precisamos. Esses programas são escritos para verificação das provaso ou não de exceções das regras *LPR*, e são operados como segue para contornarem os problemas de laços e propôr a totalidade da função computada por este, de acordo com as definições da teoria-*LPR* e o modelo definido anteriormente.

Consideramos os programas "Sleuth Inocente" e "Sleuth Culpado" simplesmente referidos daqui em diante como "Inocente" e "Culpado".

O programa "Inocente" realiza provas sobre as informações do contexto que consistem em um espaço amplo de possibilidades. Por exemplo, existem até 32 cláusulas de $nao\_match(P, X)$ que podem ser provadas por um cenário, para qualquer cenário. Além disso, suas entradas ocorrem em bem menor quatidade que as entradas do programa "Culpado". Assim sendo, iniciamos a execução pelo primeiro.

A operação dos programas, em qualquer momento como os detalhados no processo de investigação onde foram calculadas extensões anteriormente, deve seguir os passos abaixo:

1 Adicionar a "Inocente" e "Culpado" as informações obtidas externamente aos programas (e que são comuns aos dois);

2 Executar o programa "Inocente" e perguntar por provas de $no\_match(P, X)$, $match\_inocente(P)$ e $inocente(X)$.

3 Acrescentar ao programa "Culpado" as informações provadas nesta execução do programa "Inocente".

4 Executar o programa "Culpado" e perguntar por provas de $match(P, X)$, $match\_assassino(P)$ e $assassino(X)$.

5 Acrescentar ao programa "Inocente" as informações provadas nesta execução do programa "Culpado".

6 Repetir os passos [2] a [5] até que as respostas produzidas por algum

dos programas sejam iguais às respostas da execução anterior do mesmo programa.

Essa forma de operar os programas consiste, portanto, em uma montagem iterativa incremental por composição destes, de forma que:

- O conjunto de informações externas adicionadas ao programa "Inocente" consiste na base de recursão ou entradas do programa composto.

- O término da execução é indicado pela detecção de um ponto-fixo das saídas, e portanto, do operador funcional computado por esta sequência de composições de programas.

Ao fim destas operações, no momento em que um dos programas repetir seu conjunto de saídas, podemos parar, pois não haverá mais mudanças nas saídas de nenhum programa (O próprio outro programa, na execução seguinte, repetirá suas respostas, pois será executado com as mesmas entradas).

Podemos ainda argumentar que essa solução consiste em uma partição das regras que compõem um ciclo em mais que um programa, permitindo suas execuções em separado, e anulando a influência destes ciclos para o não término de sua computação. Dessa forma, tornamos as provas totais, mesmo que tais ciclos existam quando juntamos todas as operações em apenas um programa. Essa técnica divide as provas em interações discretas e com uma condição de parada bem definida, o que ocorre somente por causa da forma como o *backtracking* do prolog compõe as provas, e que provavelmente remonta ao motivo que gera a incompleteza da negação clássica na linguagem. As iterações permitem controlar o *backtracking* de uma forma alternativa ao corte, e parecem trazer muito poder expressivo ao que é oferecido no prolog.

O resultado das operações e o conjunto de fórmulas provadas consiste em um operador muito próximo de $\Psi_\tau(\Phi)$. As regras que indicam um suspeito a ser acusado neste par de programas são tais que somente apontamos aqueles que têm maior nível de culpa, e como ressaltado anteriormente, o nosso interesse neste programa é verificar de forma automatizada quando as exceções das regras LPR são provadas. Com algumas poucas modificações, podemos fazer com que os cenários sejam sempre compostos a partir de um suspeito, independente de níveis de culpa diferenciados, e produzindo as provas de exceções pertinentes a cada caso. Assim, teríamos um programa dividido em dois como aqui mostrado que, através de sucessivas operações de composição com os programas se alternando, computaria exatamente o operador $\Psi_\tau(\Phi)$.

Esta resolução por ponto-fixo é muito interessante, pois é montada sobre duas funções tecnicamente opostas, executadas sobre uma mesma base de conhecimento. Tais circunstâncias e essa descrição se assemelham às lógicas *Well-Founded*, onde são definidos dois operadores, um para afirmar o máximo que se consegue, e outro para negar o máximo que se consegue. Similarmente, esses operadores são aplicados sucessivamente, em iterações como neste modelo, e terminam em um ponto fixo da teoria. As semelhanças e possível correlação merecem uma investigação mais profunda, prevista em nossos trabalhos futuros.

Ressaltamos ainda que ao atingir o fim das operações sobre os programas e encontrar este ponto-fixo, podemos verificar por provas de cada predicado que indique níveis de culpa, e, consequentemente, teremos alguma ocorrência desses níveis para todos aqueles não tiverem sido inocentados, e uma prévia dos cenários. Ao escolher um destes e atribuir-lhe a qualidade de assassino, podemos acrescentar tal cláusula ao programa "Inocente" e buscar um novo ponto-fixo, com as mesma operações, que será então o conjunto de fórmulas provadas dentro do cenário onde este suspeito escolhido é o assassino.

Um último ponto interessante para o qual devemos chamar atenção do leitor, é que no caso de acrescentarmos, por exemplo, duas cláusulas de assassino(X) diferentes, teremos provas de que todos são inocentes, e, consequentemente, que ninguém é suspeito. Enquanto temos duas respostas analisando quem são os assassinos, não obtemos nenhuma quando verificamos quem são os suspeitos, e essa é a consulta que especifica quais cenários devemos considerar (que culpado devemos considerar em cada), neste caso, nenhum cenário a ser considerado, sobrando somente a base T. Nessa base, descrita com as informações de assassinos que supomos, temos um absurdo, pois é regra que não se pode ser assassino e inocente ao mesmo tempo, e tal condição se verificará para ambos os casos.

Da forma como aqui descritos quanto ao código e operação, esse par de programas apresenta ótimos resultados. É completamente estável e prova adequadamente os resultados a que se propõe. Claramente, a operação dos mesmos não é simples, especialmente porque a saída de cada programa não é escrita na forma de cláusulas do prolog, ou seja, a cada iteração, é necessário traduzir os resultados para que se enquadrem como entrada no programa a ser executado no passo seguinte. Além disso, é complicado observar se as respostas realmente se repetem, e utilizamos ferramentas extras para facilitar, como algum aplicativo de tabelas, onde podemos registrar por colunas as saídas de cada pergunta realizada aos programas em suas execuções.

Essas operações são complicadas, simplesmente porque a linguagem e seus interpretadores não foram desenvolvidos para esse tipo de utilização. Defendemos que um programa pode ser escrito pra interoperar os programas e fazer a tradução das saídas em entradas, e inclusive detectar o ponto-fixo. Tal programa é complexo e envolve profundo conhecimento de Prolog, e possivelmente do código de alguma implementação deste.

Capítulo 6

Conclusões e trabalhos futuros

As idéias trabalhadas na *LPR* propõem um framework lógico complexo, bastante interessante, e completo. Essa natureza das lógicas propostas em [5] não poderia ser diferente, uma vez do que estas se propõem a modelar, fornecendo ferramentas para representar todos os cenários cabíveis e entender essas coisas em uma única extensão. Essa característica única alcançada pela *LPR* é alcançada através de um processo construtivo detalhado que leva entende cada regra como um conjunto de instâncias e considerando a todas estas para calcular um conjunto maximal de instâncias compatíveis entre si. Neste conjunto maximal, obtido pelo menor ponto-fixo do operador $\Psi_\tau(Phi)$, temos uma extensão da teoria, e todos os cenários que podemos considerar para efeitos de provas. Para os cenários, utilizamos a lógica monotônica *LPD*.

Trabalhar com esse framework, e em especial, desenvolver uma aplicação utilizando a *LPR* para modelagem da teoria, foi um desafio. Como uma técnica nova, sobre a qual este é um dos primeiros trabalhos realizados, não poderia ser diferente. Tanto compreender o formalismo proposto, quanto os conceitos e cálculos de extensão propostos permitiu utilização de muitas ferramentas matemáticas e mesmo de programação, visto da especificação de um provador prolog para esta aplicação em específico.

Como resultados alcançados, temos um modelo que explora as generalizações intensivamente, mesmo que o faça através de poucas regras. Essas generalizações se combinam de forma produzir as exceções umas das outras em vários casos e permitem ilustrar o cálculo de extensões sobre uma aplicação didática e contextualizada, o que, do nosso ponto de vista, facilita o entendimento e acompanhamento das conclusões. A descrição do processo dividido em momentos também facilita ambos a contextualização, como a ilustração do funcionamento de lógicas não monotônicas no sentido de retirar uma conclusão diante de novas informações. Defendemos que, além disso, os momentos detalhados permitem analisar com calma o que ocorre no próprio processo de raciocínio do detetive em uma investigação, o que, junto ao modelo para a teoria-LPR de *Sleuth*, tratam de modelagem do conhecimento e indiciam uma abordagem a automatização do raciocínio, uma vez que empregando a possibilidade de tomar ações em um sistema com seu conhecimento modelado pela LPR e conclusões tiradas através das extensões, poderíamos orientar essas ações.

O trabalho em prolog apresentou outros desafios, especialmente porque a teoria tendia a gerar ciclos de provas. A razão disso, compreendida somente após obtenção de uma solução, é que o operador de cálculo de extensões $\Psi_\tau(Phi)$ trabalha com a noção de que as conclusões de uma regra somente fazem parte da extensão se os limites da mesma não forem provados a partir do conjunto parcial da extensão junto à própria conclusão, ou seja, a própria definição do cálculo tem um ciclo de prova, uma vez que a prova de A quando A -(B depende do próprio A. Quando tentamos realizar uma prova dessas por *backtracking*, como faz o prolog, temos uma sucessão de requisições que provocam o loop. No exemplo, a prova de A exige que não seja possível provar B, e o prolog faz uma chamada a isso. Em outras regras, temos alguma que prova B a partir de A, incluindo algumas outras condições, que modelam a possibilidade de o próprio A permitir provar B com a parte já descrita do operador de extensão. Nesse ponto, o prolog faz uma chamada pela prova de A, e temos nosso ciclo. Como este não existe em termos da lógica (quer dizer, existe, mas não gera problemas), foi bastante difícil detectar a fonte do problema e contorná-lo. Quando tratamos esse ciclo em lógica, podemos apenas verificar o que acontece se a conclusão da regra fizer parte do conjunto, e então, caso a exceção não seja provada, permitir a participação da primeira. Esse processo não perpetua o ciclo de prova indefinidamente, mas por apenas um passo, ou alguns, e é isso que conseguimos quando separamos o programa em dois. Essa técnica permite iterarmos e verificar as provas em cada passada, até que tudo o que há para ser provado pelo conjunto de fórmulas já o tenha sido, quando atingimos o ponto-fixo do nosso operador composto pelos programas “Sleuth Inocente” e “Sleuth Culpado”.

Somando a todos os fatores desenvolvidos nesse trabalho, consideramos os programas em prolog bastante satisfatórios, uma vez que nos permite observar quais cenários existem na extensão calculada com um dado conjunto de informações, bem como realizar provas sobre um cenário em específico, ou ainda, escolher algum dentre estes em que apostar baseado nas evidências coletadas até o momento. Exemplificamos, inclusive, que as conclusões de um dado momento, baseadas nesse critério, podem se mostrar errôneas, e que com um pouco mais de informação é possível tomar uma decisão mais acertada. Embora a forma que temos de operá-los seja bastante trabalhosa, estamos bastante otimistas quanto à funcionalidade do método e a possibilidade de outros usos a serem explorados.

Colocando de forma simples, este método trabalha com o prolog em duas camadas, e controla o fluxo de prova por backtracking de forma alternativa ao corte. Implementações de um interpretador para programação lógica em prolog que permita algumas novas operações, realizadas através dessa técnica de controle, ou mesmo combinações com sub-provas que não por *backtracking* parecem ser uma forte possibilidade. Assim sendo, pretendemos investigar as implicações de utilizar duas camadas de prova através de um programa único que centralize as execuções e faça a comunicação dessas partes, e a possibilidade de estender a linguagem Prolog de formas muito interessantes.

Além de testes com a prática dos dois programas, é de interesse investigar mais a fundo o potencial e/ou limites de expressividade da *LPR*, e propor outras aplicações que façam usos particularmente interessantes do cálculo de extensões e manejo de generalizações.

Como possibilidade de trabalho, também vemos a possibilidade de estender o uso deste modelo para criar um sistema autônomo de raciocínio sobre instâncias do jogo, capaz de indicar os passos ao jogador e, dessa forma, compondo um sistema completo de apoio à decisão, e, dessa forma, exemplificando a possibilidade de uso da *LPR* para criar agentes de inteligência artificial em ambientes virtuais, simulações, etc. Similarmente, vários outros jogos que, embora tenham suas instâncias criadas dedutivamente a partir de um conjunto de diretrizes, exijam algum raciocínio ampliativo para atingir a uma solução podem ser trabalhados nesse sentido. Quaisquer jogos baseados em lógica, tais como Sudoku, e jogos de tabuleiro e estratégia, também podem ser modelados e trabalhados de forma similar ao que fizemos aqui com *Sleuth*.

Referências Bibliográficas

ANTONELLI, G. Aldo; Stanford Encyclopedia of Philosophy, Dec 11, 2001, Rev Mar 27, 2006 - Non-monotonic Logic - http://plato.stanford.edu/entries/logic-nonmonotonic/3

BELL, J., MACHOVER, M.: 1977, *A Course in Mathematical Logic*. North-Holland.

BUCHSBAUM, A., BÉZIAU, J.-Y.: 2004, 'Introduction of Implication and Generalization in Axiomatic Calculi'. In: J.-Y. Béziau, A. C. Leite, and A. Facchini (eds.): *Aspects of Universal Logic*, No. 17 in Travaux de Logique. Centre de Recherches Sémiologiques, Université de Neuchâtel, p. 231.

BUCHSBAUM, A., PEQUENO, T.: 1997, 'A General Treatment for the Deduction Theorem in Open Calculi'. *Logique et Analyse* **157**, 9–29.

BUCHSBAUM, A., PEQUENO, M., PEQUENO, T.; A Logical Expression of Reasoning; Revista Synthese, 2007.

EBBINGHAUS, H-D., FLUM, J. THOMAS, W. (1994). Mathematical Logic, Springer-Verlag

ENDERTON, H, B. (1972). A Mathematical Introduction to Logic. Academic Press.

HANKS, S., MCDERMOTT, D.: 1987, 'Nonmonotonic Logic and Temporal Projection'. *Artificial Intelligence* **33**(3), 379–412.

HAWTHORNE, James; Stanford Encyclopedia of Philosophy, Sep 06, 2004 - Inductive Logic - http://plato.stanford.edu/entries/logic-inductive/

HEIT, Evan; Properties of Inductive Reasoning; Psychonomic Bulletin Review, 2000.

HENDRICKS, Vincent; Stanford Encyclopedia of Philosophy, Jan 04, 2006 - Epistemic Logic - http://plato.stanford.edu/entries/logic-epistemic/

HUGHES, G. and CRESSWELL, M., 1996, A New Introduction to Modal Logic, London: Routledge

KLEENE, S. C.: 1974, *Introduction to Metamathematics.* Wolters-Noordhoff, North Holland and American Elsevier.

KOONS, Robert; Stanford Encyclopedia of Philosophy, Jan 21, 2005 - Defeasible Reasoning - http://plato.stanford.edu/entries/reasoning-defeasible/

MCCARTHY, J.; Circumscription - A form of nonmonotonic reasoning; Artificial Intelligence **13** (1980) 27-39.

MCCARTHY, J.: 1986, 'Applications of Circumscription to Formalizing Common Sense Knowledge'. *Artificial Intelligence* **28**(1), 89–116.

MENDELSON, E.: 1979, *Introduction to Mathematical Logic.* D. Van Nostrand.

PEQUENO, M. C.; Defeasible Logic with Exception-First. PhD Thesis, Department of Computing Imperial College of Science, Technology and Medicine. 1994.

REITER, R.: 1980, 'A Logic for Default Reasoning'. *Artificial Intelligence* **13**, 81–132.

RUSSEL, S., NORVIG, P.; *Artificial Intelligence: A Modern Aproach*; Prentice Hall, 2003.

SHOENFIELD, J. R.: 1967, *Mathematical Logic.* Addison-Wesley.