PROVA DE TEOREMAS UTILIZANDO SIMPLIFICAÇÃO POR ELIMINAÇÃO DE LITERAIS

Pedro Porfírio Muniz Farias

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO

Aprovada por:

Prof^{a}. Sueli Bandeira Teixeira Mendes, Ph.D.

Prof. Roberto Lins de Carvalho, Ph.D.

Prof. Nelson Maculan Filho, Ph.D

Prof^{a}. Cláudia Maria Garcia Medeiros de Oliveira, Ph.D

Prof. Luís Alfredo Vidal de Carvalho, D.Sc.

Prof^{a}. Inês de Castro Dutra, Ph.D

RIO DE JANEIRO, RJ – BRASIL

SETEMBRO DE 1998

FARIAS, PEDRO PORFÍRIO MUNIZ

PROVA DE TEOREMAS UTILIZANDO SIMPLIFICAÇÃO POR ELIMINAÇÃO DE LITERAIS [Rio de Janeiro] 1998.

XI, 110 p. 29,7 cm (COPPE/UFRJ, D.Sc., Engenharia de Sistemas e Computação , 1998)

Tese – Universidade Federal do Rio de Janeiro , COPPE

1. Prova Automática 2. Lógica 3. Inteligência Artificial

I. COPPE/UFRJ II. Título (série)

Resumo de tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

PROVA DE TEOREMAS UTILIZANDO SIMPLIFICAÇÃO POR ELIMINAÇÃO DE LITERAIS

Pedro Porfírio Muniz Farias

Setembro/1998

Orientadores: Sueli Bandeira Teixeira Mendes, Ph.D.

Roberto Lins de Carvalho, Ph.D.

Programa: Engenharia de Sistemas e Computação

O presente trabalho apresenta um método de prova de teoremas para lógica proposicional e de primeira ordem denominado Método de Simplificação por Eliminação de Literais (MSEL). O MSEL gera conclusões a partir de fórmulas não clausais, como o método de Tableaux , evitando a geração de fórmulas redundantes que ocorre na conversão para forma clausal. Quando restringimos as fórmulas a cláusulas, o MSEL gera os mesmos resultados que a Resolução, podendo utilizar as heurísticas tradicionais empregadas para contornar a complexidade da prova a partir de bases de conhecimento. Para o caso geral, são desenvolvidas heurísticas adicionais que procuram capturar a intenção do usuário, expressa na forma escolhida para apresentar as fórmulas.

ABSTRACT OF THESIS PRESENTED TO COPPE/UFRJ AS PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF DOCTOR OF SCIENCE (D.SC.)

Pedro Porfírio Muniz Farias

September/1998

Advisors: Sueli Bandeira Teixeira Mendes, Ph.D.

Roberto Lins de Carvalho, Ph.D.

Departament: System Engineering and Computation

This work presents Method of Simplification by Literal Elimination (MSEL), a proof procedure for proposicional and first order logic theorem proving. The MSEL, like the Tableaux Method, produces conclusions from non-clausal fórmulas thereby avoiding the generation of redundant fórmulas due to the transformation of the original fórmulas into clausal form. When applied to clauses, MSEL generates the same results as Resolution, accomodating the use of the traditional heuristics used to cut down on the complexity of proofs from knowledge bases. For the general case, we present additional heuristics that capture the user's intention, expressed in the format selected to represent the fórmulas.

A MEUS PAIS, MINHA ESPOSA E FILHOS

Agradecimentos

À Professora Sueli Mendes, pela orientação, pelo grande incentivo, interesse e compreensão, pela amizade e pela influência positiva todos estes anos.

Ao Professor Roberto Lins de Carvalho, pela co-orientação, pela amizade, por compartilhar idéias a respeito deste trabalho, a respeito de ciências, de computação e de muitos assuntos que transcendem a esta tese.

Aos membros dos grupo Witty, em especial à Claudia e Alcione pela colaboração constante durante estes anos.

Aos professores da COPPE-Sistemas da linha de Inteligência Artificial, em especial a Prof.ª. Inês de Castro, pela atenção e pelas sugestões que contribuíram para melhorar este trabalho.

Aos funcionários da COPPE, pela colaboração e apoio administrativo.

Ao Banco do Nordeste e à Universidade de Fortaleza-UNIFOR pelo apoio financeiro e pela liberação de minhas atividades durante o curso.

Ao apoio institucional e financeiro do CNPQ.

A todos que, direta ou indiretamente, contribuíram para a realização deste trabalho.

ÍNDICE

1. INTRODUÇÃO

O presente capítulo inicia-se com um pequeno histórico da área, a fim de posicionar nosso estudo; segue apresentando nossas motivações e objetivos; e finaliza descrevendo a estrutura do restante do trabalho.

1.1. Histórico

Uma das maneiras mais convenientes de representar conhecimento é através de sentenças lógicas. Essa conveniência deve-se, primeiramente, à capacidade de representar adequadamente um amplo espectro de eventos. Em segundo lugar, à possibilidade de realizar inferências através de um mecanismo formalmente correto, a dedução matemática. E, finalmente, em função de a lógica e o formalismo lógico estarem arraigados em nossa cultura, na Matemática e na Filosofia.

Dada uma base de conhecimento expressa através de sentenças lógicas, não desejamos apenas recuperar as sentenças que a compõem. Nosso problema, mais amplo, é o de encontrar procedimentos para deduzir novas sentenças a partir do conhecimento contido na base.

O estudo da automatização da dedução automática de teoremas lógicos precede mesmo o desenvolvimento dos computadores. Herbrand, em 1930, expôs os fundamentos de um método para prova automática de teoremas. Em 1935, dá-se o desenvolvimento do Cálculo de Seqüentes por Gentzen.

Com base no cálculo de seqüentes, Beth criou, em 1956, o método de Tableaux semântico para lógica intuicionista (Beth,1956), que, posteriormente, em 1968, foi adaptado em (Smullyan,1968) para uso com a lógica clássica.

Nesse meio tempo, em 1960, Gilmore fez a primeira implementação do método de Herbrand e, em 1965, J. A. Robinson desenvolveu a Resolução (Robinson,1965). A Resolução representou um notável avanço no estudo da prova automática.

Na década de 70 uma série de heurísticas e variantes da resolução foram desenvolvidas. Em Lovelland,1978 encontramos uma excelente compilação dessas técnicas.

Ao final da década de 70 houve um certo declínio da área. Talvez possamos atribuir este declínio, em parte, aos resultados obtidos na área de complexidade que caracterizaram o problema da satisfabilidade para lógica proposicional como NP-completo. O problema da prova para lógica de primeira ordem tem como agravante um espaço de busca que pode ser infinito. Por outro lado, o declínio também pode ser atribuído (MacKenzie,1995) ao desapontamento em relação aos resultados práticos no período.

A partir de então, pode-se dizer que houve uma divisão que ainda hoje caracteriza a área. De um lado a proposta de construir provadores inteiramente automáticos e de outro a proposta de construir provadores interativos.

Desenvolver a prova interativamente , com a ajuda do usuário, tenderia a ser uma tarefa mais fácil. Entretanto, muitas vezes, o usuário é forçado a conhecer o problema que está sendo resolvido e ainda as características de funcionamento do provador.

Os provadores interativos são freqüentemente utilizados em problemas de verificação de programas e como auxiliares na verificação do desenho de hardware.

Pode-se subdividir os provadores segundo a lógica que implementam. Provadores de lógica de primeira ordem (com e sem igualdade), provadores para lógicas mais restritas (lógica proposicional, lógica de primeira ordem livre de quantificadores) e provadores para lógicas de mais alta ordem.

Os provadores baseados em lógicas de mais alta ordem, como o HOL (Gordon,1993), são interativos. Alguns provadores interativos, como o Isabelle (Paulson,1994), podem ser ajustados para operar com mais de uma lógica.

Os provadores para a lógica de primeira ordem constituem um classe numerosa. No CADE-15, décima quinta Conferência Internacional de Dedução Automática (Kirchner,1998) quinze concorrentes participaram do CASC-15, CADE ATP System Competition. A competição é limitada a provadores inteiramente automáticos (corretos, mas não necessariamente completos) para a lógica de primeira ordem.

No CASC são comparados os desempenhos de aplicações construídas em linguagens diferentes, rodando sob plataformas diferentes. As implementações são comparadas em relação ao número de problemas resolvidos e o tempo médio de resposta a análise é limitada a um conjunto de problemas (TPTP Program Library) e é estabelecido um limite de tempo (300 segundos) para cada tentativa de prova. Concorrem programas desenvolvidos em Lisp como o ANDP (Li,1997) com outros desenvolvidos em C , tais como o Otter (Mccune,1997).

O Gandalf (Tammet,1997) foi o sistema vencedor da competição nos anos de 1998 e 1997 e reflete, pelo menos no que se refere a de tempo de execução, o atual estado da arte. O Gandalf é implementado em Scheme e compilado com um tradutor de Scheme para C, desenvolvido por Tammet para Sun Workstation. O mecanismo de inferência utilizado é a Resolução , acrescida das estratégias de conjunto suporte, Unit-Resolution e diversas estratégias de ordenamento.

A comparação entre os sistemas existentes sob a ótica da complexidade dos algoritmos não é cabível visto que, para vários exemplos, não chegam nem mesmo a terminar a execução. Evidentemente a comparação de tempos de execução visa estimular o aparecimento de sistemas e produtos eficientes mas o resultado é influenciado pelo ambiente operacional, pela velocidade da máquina, pela linguagem e pelo compilador utilizados. Tal abordagem só reflete indiretamente a adoção de heurísticas eficientes para o tratamento do problema.

A maioria dos provadores para lógica de primeira ordem utiliza ou o Método de Tableaux ou Resolução acrescidos de um conjunto de estratégias. De um ponto de vista que desconsidere os detalhes de implementação e o ambiente de execução acreditamos que a estruturação do conhecimento, a disponibilidade de heurísticas (mais que o método) e o modo de combiná-las são os fatores que mais fortemente influenciam a eficiência da prova.

Na próxima seção comentaremos sobre o histórico do desenvolvimento dos trabalhos relacionados com a Resolução e, em seguida, retornaremos ao desenvolvimento do Tableau semântico. No capítulo 3 voltaremos aos dois métodos com maior detalhe.

1.1.1- Resolução

A Resolução é uma regra de inferência simples, já criada tendo em vista a implementação computacional, mas que só é aplicável a fórmulas traduzidas para a representação clausal.

Nas décadas de 70 e 80, a maioria dos trabalhos relacionados com prova de teoremas fez uso da Resolução.

O problema da prova automática é inerentemente complexo e de alto custo computacional. Se não forem utilizadas estratégias de busca eficientes, a prova só será viável para exemplos didáticos construídos a partir de pequenas massas de teste.

Quase todas as aplicações práticas requerem a prova de um teorema a partir de uma base de conhecimento. Nesses casos, com o crescimento do volume de dados, a questão da complexidade torna-se fundamental. Para viabilizar essas aplicações, faz-se necessária a utilização de eficientes estratégias de prova.

Visando contornar os problemas de complexidade, foram desenvolvidas várias estratégias de prova, destacando-se:

- Conjunto Suporte
- Estratégia Linear
- Estratégia Linear de Entrada
- Preferência pelas cláusulas mais curtas
- Resolução Ordenada

Apesar do grande avanço alcançado pela Resolução alguns pontos negativos devem ser ressaltados. Esses pontos negativos dizem respeito principalmente à necessidade de traduzir as fórmulas para representação clausal. A tradução para a forma clausal tem as seguintes desvantagens:

a) gera redundâncias na base de conhecimento

b) torna mais difícil explicar a dedução, uma vez que as fórmulas perderam sua forma original.

c) perde-se a intenção do usuário implícita na forma original das fórmulas. A intenção do usuário é uma valiosa informação, capaz de dar excelente suporte à elaboração de heurísticas.

Uma discussão mais detalhada desses pontos negativos é apresentada no capítulo 2.

Além dos problemas acima, em Gabbay(1993), encontramos a preocupação com o fato de as operações utilizadas para a transformação para a forma clausal não serem válidas na lógica intuicionista.

1.1.2 - Tableau Semântico

As implementações baseadas no método de prova por Tableaux tornaram-se bem menos freqüentes que aquelas baseadas em Resolução. Para isso contribuíram os seguintes principais fatores:

a) O sucesso da Resolução, um método já elaborado visando a uma implementação e a posterior pesquisa de estratégias de prova que vieram a contornar em boa parte seus problemas de eficiência;

b) o fato de o método de Tableaux ter sido apresentado, antes da primeira implementação de Gilmore, como um método gráfico a ser utilizado manualmente.

c) da forma como classicamente está exposto (Fitting,1990), sem a elaboração de estratégias equivalentes àquelas existentes para a Resolução, o método de Tableaux só é utilizável para problemas pequenos.

d) para lógica proposicional não existe o conceito de falha em um caminho de prova. Para lógica de primeira ordem, a falha está relacionada ao atingimento de um nível que representa um limite de tamanho arbitrado para a prova. A não eliminação adequada dos caminhos de prova infrutíferos gera ineficiências.

Na literatura, de uma maneira geral, o método de Tableaux é aplicado somente à verificação de tautologias.

Nos últimos anos, temos constatado crescente interesse pelo método de Tableaux. A partir de 1992 passaram a acontecer workshops anuais congregando a pesquisa em métodos de prova relacionados com Tableaux (Fronhöfer,1992), (Basin,1993), (Broda,1994), (Baungartner,1995), (Miglioli,1996), (Galmiche,1997), (Swart,1998), e, a partir de 1997, passou a ser realizado um workshop anual enfatizando provadores de teoremas para lógica de primeira ordem (http://www.logic.at/ftp98).

1.2. Motivação

Este trabalho teve como antecedentes as experiências adquiridas na análise e na utilização de uma ferramenta baseada em resolução - o Safo (Carvalho,1974), (Vieira,1985), (Vieira,1987), (Carvalho,1987), Carvalho(1988), (Menezes,1988), Carvalho(1995), (Oliveira,1995); na implementação de um provador de teoremas para

Lógica Modal utilizando Tableaux, e, finalmente, na elaboração de estudo comparativo entre os métodos de prova baseados em Tableaux e Resolução.

Nossa primeira motivação foi a de fazer as necessárias adaptações a fim de permitir a utilização de Tableaux com bases de conhecimento. Para isso seria necessária, justamente, a criação de heurísticas que pudessem tratar adequadamente a complexidade das provas.

Trabalhos recentes têm abordado essa linha de pesquisa. Em (Costa,1991) é apresentado um algoritmo para estratégia linear aplicado ao método de Tableaux clássico proposicional. Em (Avron,1993) é sugerida uma combinação de Tableau e Resolução, na qual o Tableau é utilizado para converter as fórmulas para a forma clausal, e então, a partir daí, é aplicada a Resolução.

Beckert (1995) apresenta o Lean Tableau-based Deduction, introduzindo um novo tratamento para as variáveis universalmente instanciadas.

O resultado dessa primeira fase de nossos estudos foi a criação de um método de Tableaux incompletos. A cada fórmula da base, associamos um sub-tableau no qual todos os ramos estão abertos.

Posteriormente, em face do desenvolvimento de nossos estudos, optou-se por especificar um novo método de provas, denominado: Método de Prova utilizando Simplificação por Eliminação de Literais (MSEL).

O MSEL foi implementado em Arity Prolog para o ambiente Windows 95. Os fontes estão disponíveis em www.pagina.de/porfirio.

1.3 Contribuições

Nossa primeira contribuição diz respeito aos os critérios utilizados para comparar Tableaux e Resolução: o desenvolvimento da prova; o reaproveitamento de lemas; a percepção da falha e a recuperação da busca após a falha.

Este estudo evidenciou que, para conseguir as características desejáveis da Resolução, o método de Tableaux deveria ser revisado de modo a:

a) adotar um encaminhamento goal-oriented, ou seja, incorporando, necessariamente, a estratégia de conjunto suporte. O tableaux tradicional (Smullyan,1968), (Mendes,1972), (Fitting,1990), (Fitting,1993) efetua uma busca em largura, incompatível com a complexidade do problema tratado;

b) incorporar, adicionalmente, as outras estratégias já disponíveis para resolução. Para utilizar estas estratégias, inclusive a de conjunto suporte, a Resolução produz um resolvente a partir de duas cláusulas que possuem literais de sinal complementar. No método de Tableaux , propomos estender os ramos desmembrando duas fórmulas que possuem literais com polaridade complementar.

c) modificar o mecanismo de detecção da falha – a falha no método de Tableaux ocorre apenas quando a prova atinge um comprimento maior que um limite máximo especificado. Propomos incorporar a falha em um ramo quando não é possível encontrar literais de polaridade complementar.

d) a recuperação da busca após a falha deve ser alterada, uma vez que modificamos o critério de detecção. Fazer o backtracking quando encontramos o primeiro literal isolado (para o qual não existe complementar), como na Resolução seria errado porque as fórmulas não são cláusulas. Propomos o critério de fazer o retorno a um ponto de backtracking quando constata-se que todos os literais incluídos em um ramo após este ponto de backtracking são literais isolados.

Nossa contribuição principal é o MSEL. O MSEL pode ser considerado um híbrido entre Tableaux e Resolução que incorpora , entre várias outras, as propostas acima para o método de Tableaux. Entretanto , devido a quantidade de modificações não é possível ver facilmente no MSEL os dois métodos que lhe serviram de inspiração. Assim, consideramos mais fácil explicar o MSEL caracterizando-o como um novo método.

O MSEL generaliza o método de Resolução, uma vez que pode ser aplicado a duas fórmulas quaisquer, não exigindo a tradução da fórmula para a forma clausal. O MSEL apresenta vantagens, uma vez que as deficiências apontadas na Resolução são justamente oriundas desse processo de tradução.

Nada obstante, quando aplicado a cláusulas, o MSEL funciona exatamente com os mesmos passos da Resolução.

Com relação ao Método de Tableaux, o MSEL tem como vantagens já prever a utilização de bases de conhecimento e lançar mão de adaptações das estratégias de prova disponíveis para Resolução.

Nossa terceira contribuição são as novas heurísticas apresentadas para o MSEL. A estruturação do conhecimento e a adoção de boas heurísticas são fundamentais para a prova automática bem como para qualquer problema computacionalmente difícil. As estratégias que propomos procuram capturar a intenção do usuário expressa na forma escolhida para representar as fórmulas.

Tais heurísticas utilizam informações que não estão disponíveis normalmente na Resolução, uma vez que são perdidas na transformação para forma clausal. Claro é que quanto maior a quantidade de informação que dispomos melhor tendem a ser nossas heurísticas.

É possível aplicar à Resolução heurísticas análogas as que propomos para o MSEL. Para tanto, seria necessário estender a Resolução criando estruturas de dados capazes de guardar as informações relevantes antes de fazer a conversão para a forma clausal.

1.4 Descrição do Trabalho

No Capítulo 2, são apresentadas convenções sintáticas que serão utilizadas no trabalho bem como os conceitos básicos sobre os quais se fundamenta o MSEL.

No Capítulo 3, são revistos os métodos de Resolução e de Tableaux; são estabelecidos critérios para comparar a eficiência dos métodos de prova; e comparam-se os dois métodos à luz dos critérios enunciados.

No Capítulo 4, é apresentado o MSEL proposicional, uma versão modificada com a finalidade de incorporar a regra da fatoração e a versão do MSEL para lógica de primeira ordem. Nesse capítulo, são mostradas ainda novas heurísticas que utilizam a forma das fórmulas como um critério adicional para o direcionamento das provas.

No Capítulo 5 compara-se o MSEL com outros métodos. Em especial, mostra-se que quando aplicamos o MSEL a um conjunto de cláusulas obtemos os mesmos passos que seriam obtidos se aplicássemos o método da Resolução.

No Capítulo 6 analisa-se a corretude e a completude do MSEL

O Capítulo 7 apresenta os resultados e a conclusão do trabalho.

2. CONVENÇÕES SINTÁTICAS E CONCEITOS PRELIMINARES

Este capítulo introduz convenções sintáticas e os conceitos de Negation Normal Form (NNF), polaridade (pol) e substituição com ajuste de polaridade (//) e apresenta algumas propriedades que serão necessárias nas provas de completude e corretude. Além disso vamos definir também o conceito de profundidade de um literal. A profundidade é utilizada como heurística na seleção dos literais a serem utilizados na prova.

2.1 – Convenções Sintáticas

Esta seção define alguns conceitos (Kleene,1964), (Kleene,1967), (Enderdon,1972), (Kreissel,1967) e estabelece convenções sintáticas relacionadas com a Lógica Proposicional e com a lógica de Primeira Ordem.

2.1.1 - Definições e Convenções para Lógica Proposicional

Definição 2.1

Utilizaremos como símbolos da lógica proposicional :

i) Um conjunto enumerável de símbolos proposicionais que representam proposições. Por convenção, utilizaremos letras maiúsculas A, B, C, D, ... como símbolos proposicionais;

ii) A pontuação "("e ")" para delimitar fórmulas;

iii) Os conectivos "$\vee$", "$\wedge$", "$\rightarrow$", "$\sim$", designando respectivamente a disjunção, a conjunção, a implicação e a negação;

❒

Representaremos a constante lógica verdadeiro pelo símbolo "T" e falso pelo símbolo "$\perp$".

Definição 2.2

Um literal L é um símbolo proposicional A ou sua negação ~A.

❐

Utilizaremos a letra maiúscula "L", acrescida de índices, se necessário, para representar literais.

Definição 2.3

Dois literais L_1 e L_2 são complementares se L_1 corresponde a um símbolo proposicional, "A" e L_2 a negação deste mesmo símbolo, ~A .

❐

Quando conveniente representaremos o complementar de L por ~L.

Definição 2.4

As fórmulas da lógica proposicional que, por convenção, serão representadas pelas letras gregas minúsculas, α, β, ... são obtidas indutivamente segundo as regras abaixo:

i) Todo símbolo proposicional é uma fórmula

ii) se α é uma fórmula então, $\sim\alpha$ também é uma fórmula

iii) Se α e β são fórmulas então $(\alpha \vee \beta)$, $(\alpha \wedge \beta)$, $(\alpha \rightarrow \beta)$ também são fórmulas

iv) As fórmulas são obtidas apenas por finitas aplicações das regras i) , ii) e iii) acima

❐

Em especial, quando estivermos utilizando o MSEL, a partir do capítulo 4, utilizaremos também as letras h , v e r (respectivamente, hospedeira, visitante e resultante) para representar fórmulas da lógica proposicional ou de primeira ordem, conforme o exemplo. Estas letras não serão utilizadas com nenhum outro significado.

Definição 2.5

Uma cláusula, para lógica proposicional, é uma fórmula da forma: $L_1 \vee L_2 \vee L_3 \vee \dots \vee L_n$.

❐

Utilizaremos a letra maiúscula "C", acrescida de índices, se necessário, para representar as cláusulas.

2.1.2-Definições e Convenções para Lógica de Primeira Ordem

Esta seção apresenta definições e convenções utilizadas para lógica de primeira ordem. Algumas vezes preferimos utilizar os mesmos símbolos já empregados para lógica proposicional para representar conceitos análogos presentes na lógica de primeira ordem. Nestes casos nos referiremos no texto, se necessário se fizer esclarecer, a qual das duas lógicas se refere o símbolo utilizado.

Definição 2.6

Na lógica de primeira ordem, utilizaremos como símbolos lógicos:

i) Um conjunto enumerável de variáveis. Por convenção, utilizaremos letras maiúsculas X, Y, Z, W, ... para representar as variáveis;

ii) A pontuação "(", "," e ")" para delimitar fórmulas e termos;

iii) Os conectivos "$\vee$", "$\wedge$", "$\rightarrow$", "$\sim$", designando respectivamente a disjunção, a conjunção, a implicação e a negação;

iv) O quantificador universal $\forall$e o quantificador existencial $\exists$;

e como símbolos não lógicos:

i) Um conjunto enumerável de constantes. Por convenção, utilizaremos letras minúsculas a, b, c, d, ... para representar as constantes;

ii) Um conjunto enumerável de símbolos funcionais. Representado pelas letras minúsculas f, g, ...

iii) um conjunto de símbolos predicativos. Representado pelas letras minúsculas p, q, r, ...

❒

Definição 2.7

Um termo é definido recursivamente como:

i) Uma constante é um termo;

ii) Uma variável é um termo;

iii) Se f é um símbolo funcional n-ário e t_1, t_2, ...t_n são termos, então $f(t_1,t_2,...t_n)$ é um termo;

iv) todos os termos são obtidos aplicando as regras acima.

❒

Definição 2.8

Se p é um predicado n-ário e t_1, t_2, ... t_n são termos, então $p(t_1,t_2,...t_n)$ é um átomo.

□

Definição 2.9

Um fórmula para lógica de primeira ordem, que representaremos por letras gregas minúsculas, α, β, ... é definida recursivamente , como segue:

i) Um átomo é uma fórmula;

ii) Se α é uma fórmula então $\sim\alpha$ é uma fórmula

iii) se α e β são fórmula $(\alpha\vee\beta)$, $(\alpha\rightarrow\beta)$ e $(\alpha\wedge\beta)$ são fórmulas

iv) Se α é uma fórmula e X uma variável livre em α , então $\forall(X)\ \alpha$ e $\exists(X)\alpha$ são fórmulas

v) as fórmulas são obtidas apenas por finitas aplicações das regras acima

□

Definição 2.10

Na lógica de primeira ordem, um literal L é um átomo $p(t_1,t_2,...t_n)$ ou sua negação $\sim p(t_1,t_2,...t_n)$.

□

Da mesma forma que para lógica proposicional, utilizaremos para representar literais a letra maiúscula "L", acrescida de índices, se necessário.

Definição 2.11

Dois literais L_1 e L_2 são complementares se L_1 corresponde a um átomo, $p(t_1,t_2,...t_n)$ e L_2 a negação deste mesmo átomo, $\sim p(t_1,t_2,...t_n)$.

❐

Quando conveniente representaremos o complementar de L por ~L.

Utilizaremos as definições tradicionais (Chang,1973) de substituição, skolemização e unificação. A letra grega ϕ será utilizada para representar o unificador mais geral de dois literais.

Definição 2.12

Uma cláusula, para lógica de primeira ordem, é uma fórmula da forma: $L_1 \vee L_2 \vee L_3 \vee \ldots \vee L_n$.

❐

Utilizaremos a letra maiúscula "C", acrescida de índices, se necessário, para representar as cláusulas.

Para transformar uma fórmula α num conjunto de cláusulas correspondente utilizamos o seguinte algoritmo (Chang,1973), (Casanova,1987), (Patterson,1990):

1 Verifique se a fórmula está quantificada corretamente.

Se existir alguma variável livre, ela deve ser quantificada existencialmente. Quantificadores redundantes e desnecessários devem ser eliminados. Variáveis quantificadas mais de uma vez devem ser renomeadas.

2. Elimine as implicações e mova as negações para o interior da fórmula

Transforme a fórmula substituindo as implicações pelas correspondentes disjunções. Mova as negações para o interior da fórmula, de modo que as negações só devem estar presentes junto às fórmulas atômicas.

3. Elimine os quantificadores existenciais

Introduza constantes e funções de Skolem em substituição a cada uma das variáveis quantificadas existencialmente.

4. Obtenha a forma normal prenex

Uma fórmula α está na forma normal prenex se é do tipo:

$$(Q_1X_1)\ldots(Q_nX_n)(M)\text{, onde:}$$

cada (Q_iX_i) para $i = 1\ldots n$ é $(\forall X_i)$ ou $(\exists X_i)$ e

M é uma fórmula sem quantificadores (denominada matriz de α)

Como os quantificadores existenciais já foram eliminados no passo anterior, e levando em conta a correta renomeação de variáveis efetuada no passo 1, podemos simplesmente omitir os quantificadores universais. Assim, passamos a representar a fórmula sem quantificadores subentendendo que todas as variáveis restantes estão quantificadas universalmente.

5. Obtenha a forma normal conjuntiva

Transforme a fórmula numa conjunção de disjunções.

6. Obtenha a representação clausal

Gere uma cláusula para cada uma das disjunções obtidas no passo anterior. É recomendável simplificar o resultado obtido. O procedimento de conversão para a forma clausal costuma gerar cláusulas com literais repetidos, tautologias e conjuntos de cláusulas onde umas subsumem outras.

2.2 - Negation Normal Form

Transforma-se uma fórmula para Negation Normal Form eliminando as implicações e permitindo negações somente a nível de literal. O procedimento, definido pelas regras abaixo, é uma sub-rotina do procedimento de conversão para cláusulas conjuntivas. As regras utilizadas são análogas àquelas empregadas no método de Tableaux.

O custo do procedimento é mínimo, uma vez que apenas uma passagem através da fórmula bem parentizada é necessária para completar a conversão.

Definição 2.13

Negation Normal Form de γ: $\mathcal{N}$ (γ)

A Negation Normal Form de γ, $\mathcal{N}$ (γ), é obtida aplicando-se as regras abaixo, enquanto for possível, a todas as subfórmulas α e β de γ.

Regras Tipo A

1) $\dfrac{\sim(\alpha \vee \beta)}{\sim\alpha \wedge \sim\beta}$ 2) $\dfrac{\alpha \wedge \beta}{\alpha \wedge \beta}$ 3) $\dfrac{\sim(\alpha \rightarrow \beta)}{\alpha \wedge \sim\beta}$ 4) $\dfrac{\sim\sim\alpha}{\alpha}$

Regras Tipo B

1) $\dfrac{\alpha \vee \beta}{\alpha \vee \beta}$ 2) $\dfrac{\alpha \rightarrow \beta}{\sim \alpha \ \vee \beta}$ 3) $\dfrac{\sim(\alpha \wedge \beta)}{\sim \alpha \ \vee \sim \beta}$

❐

Exemplo 2.1

A fórmula $\sim(A \rightarrow (B \rightarrow ((\sim C \vee D) \wedge E))$ foi transformada na sua correspondente em Negation Normal Form, mediante os seguintes passos:

$\sim(A \rightarrow (B \rightarrow ((\sim C \vee D) \wedge E))$	aplicando A3
$A \wedge \sim(B \rightarrow ((\sim C \vee D) \wedge E))$	aplicando A3
$A \wedge B \wedge \sim((\sim C \vee D) \wedge E))$	aplicando B3
$A \wedge B \wedge (\sim(\sim C \vee D) \vee \sim E))$	aplicando A1
$A \wedge B \wedge ((\sim\sim C \wedge \sim D) \vee \sim E))$	aplicando A4
$A \wedge B \wedge ((C \wedge \sim D) \vee \sim E))$	negation normal form

❐

Lema 2.1

A transformação de uma fórmula α para sua negation normal form: $\mathcal{N}(\alpha)$ não altera a quantidade de ocorrências de qualquer literal em α.

Prova:

Para cada ocorrência de subfórmula α ou β nos antecedentes das regras dos tipos A e B, existe uma correspondente ocorrência nos conseqüentes das regras.

A aplicação das regras dos tipos A e B não altera a quantidade de literais, mas apenas o seu sinal e o operador que os une.

❐

2.3 - Polaridade de um Literal

A Polaridade de um literal L em uma fórmula γ (pol(L)) corresponde ao sinal que o literal teria se transformássemos γ para a Negation Normal Form.

O conceito de polaridade aqui definido é análogo ao de polaridade encontrado em (Wallen,1990) e ao de ocorrência encontrado em (Ryan,1992). Segundo (Murray,1982) a noção sintática de polaridade teria sido introduzida em (Schutte,1956).

Supondo que ω é uma fórmula bem parentizada, a polaridade de cada literal é dada pela definição abaixo.

Definição 2.14

Seja p a polaridade de uma fórmula ω bem parentizada

se $\omega = \sim\alpha$ a polaridade de α é a complementar a p

se $\omega = \alpha \wedge \beta$ α e β tem polaridade p ;

se $\omega = \alpha \vee \beta$ α e β tem polaridade p;

se $\omega = \alpha \rightarrow \beta$ a polaridade de α é a complementar a p

e β tem polaridade p;

se $\omega = L$ L tem polaridade p.

❐

A polaridade de cada literal é calculada decompondo-se, indutivamente, γ em cada uma de suas subfórmulas:

Se uma fórmula γ ocorre numa base de conhecimento então sua polaridade é positiva.

Exemplo 2.2

seja : $\gamma = \omega 1 = \alpha 1 \wedge \beta 1 = (\sim A \rightarrow (\sim B \rightarrow C) \wedge \sim(D \vee E)$ polaridade +

logo :

$\alpha 1 = \omega 2 = \alpha 2 \rightarrow \beta 2 = (\sim A \rightarrow (\sim B \rightarrow C)$ polaridade +

$\alpha_2 = \omega_3 = \sim\alpha_3 = \sim A$ polaridade -

$\alpha_3 = \omega_4 = L_1 = \underline{A}$ ______ polaridade +

$\beta_1 = \omega_5 = \alpha_4 \rightarrow \beta_3 = (\sim B \rightarrow C)$ polaridade +

$\alpha_4 = \omega_6 = \sim\alpha_5 = \sim B$ polaridade -

$\alpha_5 = \omega_7 = L_2 = \underline{B}$ ______ polaridade +

$\beta_3 = \omega_8 = L_3 = \underline{C}$ ______ polaridade +

$\beta_1 = \omega_9 = \sim\alpha_6 = \sim(D \vee E)$ polaridade +

$\alpha_6 = \omega_{10} = \alpha_7 \vee \beta_4 = D \vee E$ polaridade -

$\alpha_7 = \omega_{11} = L_4 = \underline{D}$ ______ polaridade -

$\beta_4 = \omega_{12} = L_5 = \underline{E}$ ______ polaridade -

◆

Vamos indicar os sinais dos literais de uma fórmula γ com *labels* acima de cada literal. O exemplo abaixo ilustra a representação que iremos utilizar.

Exemplo 2.3

Nas fórmulas abaixo, vemos indicados as polaridades de cada literal.

$$\overset{-}{A} \rightarrow \overset{+}{B}$$

$$(\overset{+}{A} \rightarrow \overset{-}{B}) \rightarrow \overset{+}{C}$$

◆

Definimos uma função sinal (L) como a que retorna o sinal de L numa fórmula α.

Lema 2.2

Se uma fórmula α está na negation normal form, então, para todo literal L de α, temos: sinal(L) = pol(L).

Prova: Decorre diretamente da definição de polaridade.

❑

Lema 2.3

Se uma fórmula α é uma cláusula, então, para todo literal L de α, temos: sinal(L) = pol(L).

Prova: Caso particular do lema anterior, visto que toda cláusula está na negation normal form.

❑

Uma fórmula para lógica de primeira ordem livre de quantificadores (Murray,1982) é uma fórmula da lógica de primeira ordem onde as variáveis quantificadas existencialmente foram substituídas por funções de Skolem (Bhatta,1991) e os quantificadores universais foram omitidos por uma questão de facilidade sintática.

Definição 2.15

Uma fórmula para lógica de primeira ordem livre de quantificadores pode ser obtida aplicando o algoritmo abaixo (Bhatta,1991).

1 Verifique se a fórmula está quantificada corretamente.

Se existir alguma variável livre, ela deve ser quantificada existencialmente. Quantificadores redundantes e desnecessários devem ser eliminados. Variáveis quantificadas mais de uma vez devem ser renomeadas.

2. Elimine os quantificadores existenciais com polaridade positiva e os universais com polaridade negativa

Para cada quantificador existencial com polaridade positiva e cada quantificador universal com polaridade negativa, introduza constantes e funções de Skolem em substituição a cada uma das variáveis quantificadas.

3. Omita os demais quantificadores

Como os quantificadores existenciais com polaridade e positiva e os universais com polaridade negativa já foram eliminados no passo anterior, e levando em conta a correta renomeação de variáveis efetuada no passo 1, podemos simplesmente omitir os quantificadores universais com polaridade positiva e existenciais com polaridade negativa. Assim, passamos a representar a fórmula sem quantificadores subentendendo que todas as variáveis restantes estão quantificadas universalmente.

2.4 - Substituição com Ajuste de Polaridade

Se α é subfórmula de ω, a substituição com ajuste de polaridade de α por β em ω ($\omega(\alpha // \beta)$) corresponde à noção tradicional de substituição, acrescida de um ajuste do sinal de β, de acordo com o sinal e a polaridade de α.

Definição 2.16

Substituição com ajuste de polaridade - A substituição com ajuste de polaridade de α por β em ω (utilizaremos a notação $\omega(\alpha // \beta)$) é igual a:

$$\omega\ (\alpha // \beta) = \begin{Bmatrix} \omega\ (\alpha\ / \beta) & se & \text{pol}(\alpha) = \text{sinal}(\alpha) \\ \omega\ (\alpha\ / \text{complementar de } \beta) & se & \text{pol}(\alpha) \neq \text{sinal}(\alpha) \end{Bmatrix}$$

❐

Exemplo 2.5:

Sejam $\omega = (A \wedge B) \rightarrow C$, $\alpha = A$ e $\beta = D \rightarrow E$, então:

$$\omega(\alpha // \beta) = \underbrace{\overset{-}{A}}_{\overbrace{D \rightarrow E}^{-}} \wedge B \rightarrow C$$

◆

Como a polaridade negativa de $\overset{-}{A}$ é diferente de seu sinal, temos de proceder ao ajuste antes da substituição. Indicamos o ajuste de sinal na substituição através do sinal negativo acima da chave $\overbrace{D \rightarrow E}^{-}$. Essa representação é utilizada com a finalidade de tornar mais inteligíveis as substituições efetuadas.

Obtemos, finalmente:

$\omega(\alpha // \beta) = (\sim(D \rightarrow E) \wedge B) \rightarrow C$

Exemplo 2.6:

Sejam $\omega = A \rightarrow (B \vee C)$, $\alpha = B$ e $\beta = D \rightarrow E$, então:

$$\omega(\alpha // \beta) = A \rightarrow \underbrace{\overset{+}{B}}_{\overbrace{D \rightarrow E}} \vee C$$

◆

Como a polaridade de $\overset{+}{B}$ é igual a seu sinal, o ajuste não se faz necessário. Temos então:

$\omega(\alpha // \beta) = \omega(\alpha / \beta) = A \rightarrow ((D \rightarrow E) \vee C)$

Lema 2.4

Se a fórmula ω está na negation normal form, então, para todo literal L de ω, $\omega\,(L\,//\,\beta) = \omega\,(L\,/\,\beta)$.

Prova: Como, para todo literal de ω, pol(L)=sinal(L), logo, não são necessários ajustes de sinais para fórmulas na negation normal form.

❑

Lema 2.5

Se uma cláusula C está na forma normal conjuntiva disjuntiva, então, para todo literal L de C, $C\,(L\,//\,\beta) = C\,(L\,/\,\beta)$.

Prova: Segue do fato de a forma normal conjuntiva disjuntiva ser um caso particular da negation normal form.

❑

Lema 2.6

$$\mathcal{N}\,(\omega\,(L_1\,//\,L_2)\,) = \mathcal{N}\,(\omega\,)\,(L_1\,/\,L_2)$$

Prova: Decorre do Lema 1 e da definição de substituição com ajuste de sinal. A conversão para negation normal form preserva as ocorrências de literais na fórmula ω modificando apenas os sinais destes literais. Ao converter a fórmula para $\mathcal{N}\,(\omega\,)$ o literal L_1 passa a ter pol(L_1)=sinal(L_1), logo não há necessidade de ajuste de sinal durante a substituição do literal L_1 pelo literal L_2.

❑

Exemplo 2.7

Seja $\omega = (A \rightarrow (B \wedge C)) \rightarrow D$, onde desejamos substituir a ocorrência negativa de B por uma ocorrência negativa do literal E. Então:

$$\omega(\bar{B} \mathbin{//} {\sim}E) = \omega(B/E) = (A \rightarrow (E \wedge C)) \rightarrow D$$

A substituição exige um ajuste de sinal, uma vez que pol(B) $\neq$ sinal(B). Transformando o resultado para negation normal form, temos:

$$\mathcal{N}(\omega(B \mathbin{//} {\sim}E)) = (A \wedge ({\sim}E \vee {\sim}C)) \vee D$$

Por outro lado, obtemos o mesmo resultado fazendo primeiro a conversão para $\mathcal{N}(\omega)$, e, em seguida, a substituição da ocorrência negativa de B pela ocorrência negativa de E:

$$\mathcal{N}(\omega) = (A \wedge ({\sim}B \vee {\sim}C)) \vee D$$

$$\mathcal{N}(\omega)(\bar{B} \mathbin{//} {\sim}E) = (A \wedge ({\sim}E \vee {\sim}C)) \vee D = \mathcal{N}(\omega(B \mathbin{//} {\sim}E))$$

◆

Lema 2.7

$$\mathcal{N}(\omega(L_I \mathbin{//} \beta)) = \mathcal{N}(\omega)(L_I / \mathcal{N}(\beta))$$

Prova: Análoga ao Lema anterior. A diferença consiste em substituir L_I por uma fórmula, em vez de substituí-lo por outro literal. No segundo lado da igualdade, é preciso converter β para negation normal form antes de efetuar a substituição. Esse cuidado é necessário pois $\mathcal{N}(\omega)(L_I / \beta)$ só está na negation normal form, se β estiver.

❑

Exemplo 2.8

Seja $\omega = (A \rightarrow (B \wedge C)) \rightarrow D$, onde desejamos substituir a ocorrência negativa de B pela fórmula ${\sim}E \rightarrow F$. Então:

$\omega(\bar{B} // \sim E \to F) = \omega(B / \sim(\sim E \to F)) = (A \to (\sim(\sim E \to F) \wedge C)) \to D$

A substituição exige um ajuste de sinal, uma vez que pol(B) $\neq$ sinal(B). Transformando o resultado para negation normal form, temos:

$\mathcal{N}(\omega(B // \sim E \to F)) = (A \wedge (E \vee \sim F \vee \sim C)) \vee D$

Por outro lado, obtemos o mesmo resultado fazendo primeiro a conversão para $\mathcal{N}(\omega)$, e, em seguida, a substituição da ocorrência negativa de B pela negation normal form de $\sim E \to F$:

$\mathcal{N}(\omega) = (A \wedge (\sim B \vee \sim C)) \vee D$

$\mathcal{N}(\sim E \to F) = E \vee \sim F$

$\mathcal{N}(\omega)(\bar{B} // \mathcal{N}(\sim E \to F)) = (A \wedge (E \vee \sim F \vee \sim C)) \vee D = \mathcal{N}(\omega(B // \sim E \to F))$

◆

2.5 - Profundidade de um Literal

A profundidade de um literal L em uma fórmula γ corresponde ao nível de aninhamento de disjunções, onde o literal L apareceria, se escrevêssemos γ sob a Negation Normal Form.

O cálculo da profundidade de um literal pode e deve ser feito concomitantemente com o cálculo de seu sinal.

Definição 2.17

Profundidade disjuntiva de um Literal L

A profundidade de um literal L pertencente a uma fórmula α, P(α,L) é dada pelo número de níveis de alternativas (conjunções não contam) que se aninhariam sob L, caso α estivesse na Negation Normal Form.

P(α,L) pode ser definida, indutivamente, pela regra abaixo:

$$p(\alpha, L) = \begin{cases} 0 & se \quad \alpha = L \\ p(\alpha', L) & se \quad \alpha = \alpha' \wedge \ldots \wedge \alpha^{n'} \quad e \quad L \in \alpha' \\ p(\alpha', L) + 1 & se \quad \alpha = \alpha' \vee \ldots \vee \alpha^{n'} \quad e \quad L \in \alpha' \end{cases}$$

❐

Exemplo 2.9

Nos exemplos que seguem, os *labels* acima de cada literal indicam a profundidade

a) $\overset{0}{A}$

b) $\overset{0}{A} \wedge \overset{0}{B}$

c) $\overset{1}{A} \rightarrow \overset{1}{B}$

d) $((\overset{1}{A} \rightarrow (\overset{1}{B} \vee \overset{1}{C} \vee (\overset{2}{D} \wedge \overset{2}{E}))) \rightarrow \overset{1}{F}) \vee \overset{1}{G}$ pois, na Negation Normal Form, temos:

$$(\overset{1}{A} \wedge \sim \overset{1}{B} \wedge \sim \overset{1}{C} \wedge (\sim \overset{2}{D} \vee \sim \overset{2}{E})) \vee \overset{1}{F} \vee \overset{1}{G}$$

ou, reescrevendo a fórmula acima numa representação que evidencie os níveis:

$$(\overset{1}{A} \wedge \sim \overset{1}{B} \wedge \sim \overset{1}{C} \wedge \begin{pmatrix} \\ \sim \overset{2}{D} \vee \sim \overset{2}{E} \end{pmatrix}) \vee \overset{1}{F} \vee \overset{1}{G}$$

◆

A profundidade de uma fórmula é igual à de seu literal de maior profundidade.

A grande maioria das fórmulas que o usuário escreve tem profundidade 0 (fatos) ou 1 (regras simples). As fórmulas de profundidade ≥ 2 são exceção em qualquer base de conhecimento.

Nas heurísticas que desenvolveremos, vamos dar preferência à utilização de literais com menor profundidade, e, como segundo critério, à utilização de fórmulas com menor profundidade.

3. TABLEAUX E RESOLUÇÃO

No presente Capítulo é feita uma apresentação sucinta dos métodos de Tableaux e Resolução, enfatizando alguns pontos que nos motivaram e que serão particularmente úteis no MSEL. Em seguida, vamos determinar critérios para análise da eficiência dos Métodos de Prova e vamos comparar os Métodos de Tableaux e Resolução segundo os critérios selecionados.

3.1 - Resolução

O método de prova baseado em Resolução (Robinson,1965) é extensivamente analisado na literatura. Para aplicá-lo, precisamos, primeiramente, transformar cada fórmula num conjunto de cláusulas na forma normal conjuntiva (CNF). Em seguida, dadas duas cláusulas C_1 e C_2, a resolução permite deduzir uma outra cláusula, R, denominada resolvente de C_1 e C_2.

3.1.1 - Problemas decorrentes da transformação para forma clausal

A transformação para a forma clausal tem como vantagem a subdivisão do conhecimento expresso nas fórmulas em subfórmulas mais simples e homogêneas. A simplificação da representação redunda numa simplificação do cálculo do resolvente.

Entretanto, como já havia sido enfatizado na introdução deste trabalho, a transformação para forma clausal acarreta as seguintes desvantagens:

a) gera redundâncias na base de conhecimento. O exemplo 3.1 mostra um caso em que a geração de redundâncias é propositadamente alta.

b) torna mais difícil explicar a dedução, uma vez que as fórmulas perdem sua forma original.

c) perde-se a intenção do usuário, implícita na forma original da fórmula. A intenção do usuário é uma valiosa informação capaz de dar excelente suporte à elaboração de heurísticas. O Exemplo 3.2 mostra uma estratégia que utiliza a posição dos literais dentro da implicação (antecedente ou conseqüente) como critério para escolha do melhor caminho de prova.

Exemplo 3.1

A transformação da fórmula $((A \rightarrow B) \wedge (C \rightarrow D) \wedge (A \rightarrow A)) \rightarrow A$ gera as oito cláusulas abaixo :

1)	$A \vee C \vee A \vee A$	5)	$A \vee C \vee {\sim}A \vee A$
2)	${\sim}B \vee C \vee A \vee A$	6)	$B \vee C \vee {\sim}A \vee A$
3)	$A \vee {\sim}D \vee A \vee A$	7)	$A \vee {\sim}D \vee {\sim}A \vee A$
4)	${\sim}B \vee {\sim}D \vee A \vee A$	8)	${\sim}B \vee {\sim}D \vee {\sim}A \vee A$

◆

Vale notar que quatro das cláusulas acima (5, 6, 7, e 8) são tautologias. A cláusula 3) subsume a cláusula 4) e a cláusula 1) subsume a cláusula 2).

A informação redundante pode ser diminuída se eliminarmos as tautologias e verificarmos as subsunções. Esses procedimentos são onerosos e só são aplicáveis quando da criação da base. Se forem geradas no decorrer da prova, as tautologias e as cláusulas que podem gerar subsunções provocam um acréscimo desnecessário no esforço da prova.

Em (Avron,1993) é proposta a utilização de Tableaux na conversão para a forma clausal. Dessa forma, a geração de tautologias, mas não as subsunções, poderia ser eliminada através do fechamento dos ramos nos quais as tautologias estivessem presentes.

Entretanto, sempre permanecerá a redundância decorrente da distribuição da conjunção sobre a disjunção. Como na fórmula $C \vee (A \wedge \sim B)$ gerando as cláusulas $C \vee A$ e $C \vee \sim B$. O trabalho de resolver o literal 'C' pode ser duplicado, uma vez que ele foi distribuído nas duas cláusulas.

Vamos agora exemplificar como a conversão para forma clausal oculta a intenção do usuário.

Evidentemente, qualquer tentativa de capturar a intenção do usuário deve ser vista como uma questão de heurística. Estamos supondo que duas fórmulas logicamente equivalentes escritas de maneiras diferentes refletem intenções diferentes.

Assim, quando o usuário escreve $A \rightarrow B$ a intenção é diferente de quando escreve $\sim A \vee B$. A forma escolhida pelo usuário para apresentar a fórmula pode servir de base para uma série de heurísticas.

Exemplo 3.2

Desejamos provar $A \rightarrow D$ a partir da base $\Gamma=\{ A \rightarrow B, B \rightarrow C, C \rightarrow D, C \vee E, \sim B \rightarrow \sim F, \sim D \rightarrow G\}$. Abaixo, temos a conversão de cada fórmula para forma clausal:

	fórmula	*cláusula*
1)	$A \rightarrow B$	$\sim A \vee B$
2)	$B \rightarrow C$	$\sim B \vee C$
3)	$C \rightarrow D$	$\sim C \vee D$
4)	$C \vee E$	$C \vee E$
5)	$\sim B \rightarrow \sim F$	$B \vee \sim F$
6)	$\sim D \rightarrow G$	$D \vee G$
G1)	$\sim(A \rightarrow D)$	$\sim D$
G2)		A

◆

Quando tentamos provar que $A \rightarrow D$, que 'D' é conseqüência de 'A', temos conhecimento de que 'A' é um antecedente e que 'D' é um conseqüente. Essa informação fundamental para a condução da prova é perdida na tradução para forma clausal. Com base nela, podemos escolher dois direcionamentos para a prova. O primeiro, partindo dos conseqüentes e procurando chegar aos antecedentes, estratégia que vamos denominar "primeiro os consequentes"; e o segundo direcionamento, partindo dos antecedentes e tentando chegar aos conseqüentes, que, analogamente, vamos denominar "primeiro os antecedentes".

Observando a cláusula G1), não temos nada que indique que '~D' é conseqüente. Temos a opção de resolver o literal '~D' de G1 contra as cláusulas 3) ou 6). A transformação da cláusula 3) para a forma clausal nos fez perder a informação de que 'D' na fórmula 3) era um conseqüente. Da mesma forma, a transformação da cláusula 6) nos fez perder a informação de que '~D' na fórmula 6) era um antecedente. Perdemos todas as informações relevantes para decidir entre a cláusula 3) e a cláusula 6).

Ora, desejamos provar que entre as conseqüências de 'A' temos o literal 'D'. Se a intenção tivesse sido preservada, deveríamos preferir fórmulas onde 'D' ocorre como conseqüente, e então descobriríamos se 'A' está entre os antecedentes que permitem concluir 'D'. Se ainda soubéssemos que o literal '~D' de G1) é um conseqüente, a fórmula 3) deveria ser escolhida. Teríamos, então:

R1) ~C de G1) e 3)

Seguindo o direcionamento, desejamos saber que antecedentes permitem concluir 'C'. Temos a opção de resolver R1) com 2) e 4) mas, novamente, a conversão para forma clausal nos fez perder a informação da qual necessitávamos para direcionar corretamente a busca. A escolha acertada seria 2) onde 'C' é conseqüente. Teríamos, então:

R2) ~B e R1) e 2)

Agora, podemos resolver R2) com a cláusula 1) ou com a cláusula 5). Novamente a conversão para forma clausal dificulta a escolha correta. Devemos preferir a cláusula 1) onde 'b' é conseqüente. Finalmente, temos:

R3) ~A de R2) e 1)
R4) ⊥ de R3) e G2)

No Safo (Carvalho,1995) a intenção procura ser capturada através da construção de uma hierarquia de conceitos. Essa hierarquia é registrada numa estrutura de dados adicional, denominada grafo semântico, que não está disponível em outros provadores, nem está prevista na literatura relacionada com Resolução. O grafo semântico é construído antes da conversão para a forma clausal e registra informações que serão perdidas no momento da conversão.

Além dos problemas acima exemplificados, em (Gabbay,1993), como anteriormente mencionado,encontramos a preocupação com o fato de as operações utilizadas para a transformação para a forma clausal não serem válidas na lógica intuicionista. Os problemas existentes na transformação para forma clausal são utilizados como argumento para a criação de um método de prova "goal-oriented".

3.1.2 - O Princípio da Resolução

Feita a transformação para forma clausal, a dedução se faz segundo o princípio da resolução.

Definição 3.1

Princípio da Resolução - *Para quaisquer duas cláusulas C_1 e C_2. Se existe um literal L_1 em C_1 que é complementar a um literal L_2 de C_2, então elimine L_1 e L_2 de C_1 e C_2, respectivamente, e construa a disjunção das cláusulas remanescentes. A cláusula construída é o resolvente 'binário' de C_1 e C_2 (Chang,1973).*

❐

Para que duas cláusulas C_1 e C_2 sejam resolvidas, devem existir um literal L_1 em C_1 e um literal L_2 em C_2 , tais que existe uma substituição ϕ (m.g.u) que aplicada às variáveis de L_1 e L_2 os torna literais complementares. Ou seja, $L_1\phi = \sim L_2\phi$.

Obtém-se a cláusula resolvente $R = (C_1\phi - L_1\phi) \vee (C_2\phi - L_2\phi)$, aplicando a substituição ϕ às cláusulas C_1 e C_2, eliminando $L_1\phi$ de $C_1\phi$, eliminando $L_2\phi$ de $C_2\phi$ e, finalmente, fazendo a disjunção das duas cláusulas remanescentes.

Exemplo 3.3

$$\begin{tikzcd}[column sep=small]
p(X) \vee \sim q(X) \arrow[dr] & & q(a) \vee r(Z) \arrow[dl] \\
& p(a) \vee r(Z) &
\end{tikzcd}$$

◆

3.1.3 -Resolução e Fatoração

A inferência através da resolução tem que ser complementada pela fatoração. A fatoração é necessária para atender aos problemas resultantes da duplicação de literais dentro da mesma cláusula.

No exemplo 3.4 mostra-se uma prova impossível de ser realizada somente com a resolução e sem contar com a fatoração.

Exemplo 3.4

1)	$A \vee A$	
2)	$\sim A \vee \sim A$	
3)	$A \vee \sim A$	resolvendo 1) e 2)
4)	$A \vee A$	resolvendo 3) e 1)
		loop

◆

A fatoração permite descartar as ocorrências repetidas de um mesmo literal dentro de uma cláusula, apresentando como resultado uma única instância desse literal.

Utilizando a fatoração, teríamos, para o exemplo 3.4:

3)	A	fatorando 1)
4)	$\sim A$	fatorando 2)
5)	$\perp$	resolvendo 3) e 4)

Definição 3.2

Se existem, numa cláusula C, dois ou mais literais L_1, L_2, L_3, ...,L_n com o mesmo sinal e para os quais existe um mais geral unificador θ ($L_1\theta = L_2\theta = L_3\theta = \ldots = L_n\theta$), então Cθ é denominado fator de C. Onde Cθ é a aplicação do unificador θ sobre a cláusula C, eliminando-se os literais duplicados L_2, L_3, ...,L_n (Casanova,1987).

❐

Exemplo 3.5

Para $C = p(a) \vee q(b,X) \vee p(X)$ temos o fator de C:

$C\theta = p(a) \vee q(b,a)$ onde, $\theta = \{X/a\}$.

◆

Algumas abordagens apresentam a fatoração como uma segunda regra de inferência (Loveland,1978), outras como (Chang,1973) e (Casanova,1987) preferem incluir a fatoração na definição do resolvente.

Definição 3.3

Um resolvente de duas cláusulas C_1 e C_2 é um resolvente binário de C_1 (ou um fator de C_1) com C_2 (ou um fator de C_2).

❐

Exemplo 3.6

$$\begin{tikzcd} p(a,X) \vee q(X) \arrow[ddr] & & \sim p(Y,b) \vee \sim p(a,b) \arrow[d, "\text{por fatoração}"] \\ & & \sim p(a,b) \arrow[dl] \\ & q(b) & \end{tikzcd}$$

◆

A fatoração deve sempre preceder a resolução. Todavia, no cálculo de primeira ordem, a aplicação da fatoração pode ser indevida. É necessário portanto o cuidado de introduzir um ponto de *backtracking* todas as vezes em que se aplica a fatoração. Esse detalhe é freqüentemente omitido na literatura (Casanova,1987),(Chang,1973), principalmente porque a fatoração é apresentada embutida na definição do resolvente.

O exemplo 3.7 mostra um caso em que é necessário desfazer a fatoração através do *backtracking*.

Exemplo 3.7

1)	p(a,X) ∨ p(Y,b)	
2)	~p(d,b)	
3)	~p(a,c)	
4)	p(a,b)	fatorando 1)
		falha e faz backtracking
5)	p(Y,b)	resolvendo 1) e 3)
6)	⊥	resolvendo 5) e 2)

◆

Uma alternativa para a utilização da fatoração é a utilização do método da Eliminação de Modelos (Loveland,1978). No método da eliminação de modelos, a regra de redução corresponde à fatoração. De qualquer forma, tanto na fatoração, quanto na redução, o ponto de backtracking deve ser incluído.

3.1.4 - O Procedimento de Prova

Com base na resolução, o procedimento de prova mais elementar é gerar sistematicamente todas as cláusulas possíveis, verificando, a cada passo, se é gerada a cláusula vazia. Tal procedimento, conhecido como resolução por saturação, pode ser definido pelo procedimento abaixo:

Definição 3.4

Procedimento de Resolução por Saturação

Seja $\{\alpha_1, \alpha_2, \alpha_3, \ldots..\alpha_n\}$ o conjunto de fórmulas que compõe uma base de conhecimento

Seja Γ o conjunto de cláusulas geradas a partir de $\{\alpha_1, \alpha_2, \alpha_3, \ldots..\alpha_n\}$

Seja γ a fórmula objetivo para a qual se deseja saber se $\alpha_1, \alpha_2, \alpha_3, \ldots..\alpha_n \vdash \gamma$,

Seja Δ o conjunto de cláusulas geradas a partir de $\sim\gamma$.

Define-se a seqüência $\Gamma_0, \Gamma_1, \Gamma_2, \ldots, \Gamma_n$, onde:

$\Gamma_0 = \Gamma \cup \Delta$,o conjunto inicial de cláusulas

REPITA

$\Gamma_{n+1} = \Gamma_n \cup \{C_R / C_R$ é resolvente de duas cláusulas de $\Gamma_n\}$.

SE a cláusula vazia for gerada

ENTÃO pare e exiba 'PROVOU α a partir de Γ'.

ATÉ QUE não seja possível gerar uma nova cláusula,

pare e exiba 'Não é possível provar α a partir de Γ'.

❐

O procedimento de prova de resolução por saturação é refutacionalmente correto e refutacionalmente completo (se $\Gamma \cup \Delta$ é insatisfatível, então a cláusula vazia será gerada). Entretanto, como qualquer outro método aplicado a fórmulas da lógica de primeira ordem, é apenas um procedimento de decisão parcial, uma vez que só termina e responde negativamente quando o conjunto de resolventes é finito.

Para se lidar com o ciclo nos casos em que é possível gerar infinitamente novos resolventes, arbitra-se um limite a partir do qual desistimos da tentativa de prova.

3.1.5 - Estratégias de Prova

É importante perceber que, à medida que cresce o número de cláusulas, as combinações que podem ser geradas aumentam exponencialmente. Assim sendo, lidar com essa explosão computacional, estabeleceu-se uma série de heurísticas que visam determinar uma prova de modo mais eficiente.

Agrupamos (Chang,1973) essas estratégias em três tipos: Estratégias de Simplificação, Estratégias de Refinamento e Estratégias de Ordenação.

3.1.5.1- Estratégias de Simplificação

Consistem na eliminação de certas cláusulas ou de certos literais dentro das cláusulas. A simplificação é feita de modo que, se o conjunto inicial de cláusulas é insatisfatível, o conjunto de cláusulas resultante da simplificação também será insatisfatível.

a) Eliminação de Tautologias

As cláusulas que contêm tautologias não devem ser consideradas, uma vez que qualquer conjunto insatisfatível contendo a tautologia também será insatisfatível se a excluirmos.

Um cuidado especial deve ser tomado quando da conversão das fórmulas para a Forma Normal Conjuntiva. Nesse processo, freqüentemente são criadas cláusulas contendo tautologias que podem ser descartadas antes mesmo de serem inseridas na base de conhecimento.

b) Eliminação por Subsunção

Uma cláusula C_1 subsume outra C_2 se existe uma substituição ϕ tal que $c_1\phi \subseteq c_2$. As cláusulas subsumidas podem ser eliminadas.

O processo de conversão para Forma Normal Conjuntiva também gera muitas cláusulas que podem ser eliminadas segundo o critério da subsunção.

c) Eliminação de Cláusulas com literais isolados (pure-literal)

Considera-se literal isolado aquele que não tem um outro literal complementar na base. A cláusula C_1 possui um literal isolado L_1 se não existe numa cláusula C_2 (igual ou diferente de C_1) o literal L_2, tal que existe uma substituição ϕ (m.g.u) que aplicada às variáveis de L_1 e L_2 os torne literais complementares.

A rigor, deve-se exigir ainda que C_2, por sua vez, não tenha um outro literal isolado L_3, pois nesse caso, C_2 deveria ter sido retirada da base e portanto seus literais não poderiam estar sendo comparados com os de C_1.

Por uma questão de custo, podem-se relaxar as exigências acima, esquecendo de observar se L_1 e L_2 unificam, e considerando isolados somente os literais para os quais não existe na base um outro literal com sinal oposto.

Evidentemente, as cláusulas que possuem literais isolados não poderão ser resolvidas totalmente, e, portanto, não deveriam ser usadas na refutação.

O sentido de aceitar a existência de literais isolados numa base de cláusulas é que eles provavelmente resolverão contra as perguntas solicitadas à base.

Assim sendo, efetuada a negação da pergunta e iniciada a refutação, deveriam ser eliminadas da base as cláusulas que ainda possuíssem literais isolados. Outra alternativa seria a de somente permitir a resolução de cláusulas isoladas contra a pergunta, inibindo as outras resoluções possíveis.

3.1.5.2 - Estratégias de Refinamento

As estratégias de refinamento limitam o espaço de busca da prova, restringindo as cláusulas que são passíveis de serem resolvidas a cada passo. Esse tipo de restrição, apesar de fundamental, pode comprometer a completude refutacional do método .

As estratégias de refinamento mais freqüentemente utilizadas são descritas nas subseções seguintes.

a) Conjunto Suporte

A estratégia do conjunto suporte parte da suposição de que a base de conhecimento, Γ, é consistente, e de que a inclusão das cláusulas referentes à negação da fórmula objetivo, Δ, tornará $\Gamma \cup \Delta$ inconsistente.

Sob essa hipótese, nenhuma resolução entre duas cláusulas de Γ pode gerar a cláusula vazia. Adota-se, portanto, a restrição de não permitir resolução entre as cláusulas de Γ.

Essa restrição é muito eficiente, visto que Γ tende a ter um número de cláusulas bem maior que Δ. E é tão mais eficiente quanto maior for o número de cláusulas de Γ. A estratégia de conjunto suporte é absolutamente essencial quando se pretende tratar com bases de conhecimento que não sejam muito pequenas.

Procura-se, portanto, direcionar a busca no sentido de descobrir quais cláusulas do conjunto Δ provocam a inconsistência de $\Gamma \cup \Delta$. Esse direcionamento é bastante intuitivo tendo em vista que Δ está associado a uma consulta.

Para se implementar a restrição de não permitir a resolução entre cláusulas de Γ, modifica-se o procedimento de resolução por saturação definindo-se o conjunto $\Delta_0 = \Delta$, denominado conjunto suporte inicial. O conjunto suporte, Δ_i, é progressivamente, aumentado acrescendo-se o resolvente obtido a cada passo da prova.

Temos, então, $\Delta_{i+1}=\Delta_i \cup \{C\}$, onde C é resolvente de uma cláusula de Δ_i e outra cláusula de Γ, ou C é resolvente de duas cláusulas de Δ_i. Os critérios de parada são os mesmos da resolução por saturação: se é gerada a cláusula vazia, o método responde afirmativamente, e, se não há mais cláusulas a serem geradas, o método pára e responde negativamente.

b) Estratégia Linear

A estratégia linear corresponde a uma busca em profundidade, na qual o resolvente de cada passo é utilizado no passo seguinte como uma das cláusulas que serão resolvidas.

A estratégia linear é completa, e freqüentemente é apresentada associada com o método do conjunto suporte. A associação dos dois métodos também resulta numa estratégia completa.

c) Estratégia linear com restrição de entrada

A estratégia linear exige que uma das cláusulas a serem resolvidas seja o resolvente do passo de prova anterior. A restrição de entrada exige, adicionalmente, que a outra cláusula a ser resolvida pertença à base de conhecimento Γ.

A estratégia linear com restrição de entrada não é completa. Por vezes é necessário resolver duas cláusulas que pertençam ao conjunto suporte. Esse inconveniente pode ser minimizado transformando-se a restrição numa preferência. A estratégia linear seria combinada com uma estratégia de ordenação onde as cláusulas da base de conhecimento teriam preferência em relação às do conjunto suporte.

3.1.5.3 - Estratégias de Ordenação

Muito embora a Resolução não especifique nenhuma ordenação em particular, normalmente, pelo menos por razões de implementação, as cláusulas são selecionadas na ordem em que foram escritas, e os literais, da esquerda para direita. Todavia, é comum instituir-se uma função de prioridade na seleção dos literais que vão ser resolvidos.

As estratégias de ordenação não proíbem qualquer resolução, apenas fornecem uma heurística a respeito de quais devem ser executadas em primeiro lugar. Assim sendo, o acréscimo de heurísticas de ordenação não altera a completude de qualquer método.

São exemplos de heurísticas de ordenação:

a) Ciranda de Fórmulas - Consiste em postergar ao máximo a reutilização de uma cláusula já utilizada na prova. Mais que uma heurística, este é um procedimento obrigatório. Sem ele, a maior parte das provas entraria em ciclo rapidamente. (Carvalho, 1995), (Fitting,1990).

b) Preferência pelas mais curtas - Supõe-se que resolvendo as cláusulas mais curtas chegaríamos mais rapidamente à cláusula vazia. De fato, para que a cláusula vazia seja gerada, necessariamente cláusulas cada vez mais curtas devem ser utilizadas, até que, ao final, duas cláusulas unitárias devem ser resolvidas.

c) Preferência pelas unitárias - Consiste em priorizar os fatos, ou cláusulas unitárias, em relação às regras. É um caso particular da preferência pelas mais curtas.

É possível gerar heurísticas mais eficientes, procurando-se capturar a intenção registrada na forma escolhida pelo usuário para apresentar as fórmulas. Já nos referimos a heurística de preferência pelos conseqüentes, que leva em conta o posicionamento do literal com relação às implicações possíveis. No Capítulo 4 apresenta-se outra heurística de ordenação baseada num nível atribuído a cada literal.

O SAFO (Carvalho,1995), (Viera,1985) além do rodízio de fórmulas e das estratégias de preferência pelas mais curtas e pelas cláusulas unitárias, utiliza ainda uma estratégia de ordenação, através de um grafo semântico que é gerado com base no exame das fórmulas antes da conversão para forma clausal.

3.2 - *Tableaux*

O método de prova usando refutação através de Tableaux foi apresentado por Beth (1964) e reformulado sob uma notação uniforme por Smullyan (1968). Apesar de tradicionalmente ser apresentado como um método refutacional, o método de Tableaux também pode ser empregado como método de prova direta (Bibel,1993). Esta seção apresenta o método de Tableaux para lógica proposicional e para lógica de primeira ordem (Mendes,1972), e indica como o método pode ser utilizado para prova direta.

3.2.1 - Tableau Proposicional

Para provar uma fórmula ω, o método de Tableaux procura refutar a fórmula $\sim\omega$, criando uma árvore que representa a forma disjuntiva de $\sim\omega$. A árvore é construída decompondo $\sim\omega$ nas suas subfórmulas através das regras α, β, abaixo. Um ramo é criado para cada disjunção de $\sim\omega$, e cada ramo individualmente representa a conjunção dos literais que o compõem.

Definição 3.5

Denomina-se desenvolvimento de uma fórmula ψ a decomposição de ψ em suas subfórmulas, segundo as regras do tipo α e do tipo β, abaixo.

Regras Tipo α:

$$\begin{array}{c} \omega \wedge \varphi \\ \hline \omega \\ \varphi \end{array} \qquad \begin{array}{c} \sim(\omega \rightarrow \varphi) \\ \hline \omega \\ \sim\varphi \end{array} \qquad \begin{array}{c} \sim(\omega \vee \varphi) \\ \hline \sim\omega \\ \sim\varphi \end{array} \qquad \begin{array}{c} \sim\sim\omega \\ \hline \omega \\ \omega \end{array}$$ [1]

Regras Tipo β:

$$\begin{array}{c|c} \multicolumn{2}{c}{\omega \vee \varphi} \\ \hline \omega & \varphi \end{array} \qquad \begin{array}{c|c} \multicolumn{2}{c}{\omega \rightarrow \varphi} \\ \hline \sim\omega & \varphi \end{array} \qquad \begin{array}{c|c} \multicolumn{2}{c}{\sim(\omega \wedge \varphi)} \\ \hline \sim\omega & \sim\varphi \end{array}$$

❐

Diz-se que uma fórmula é do tipo α se pode ser desenvolvida por uma regra do tipo α; e diz-se que uma fórmula é do tipo β se pode ser desenvolvida por uma regra do tipo β. Vamos nos referir às regras do tipo α e do tipo β, de maneira sintética, como:

$$\begin{array}{c|c} \multicolumn{2}{c}{\beta} \\ \hline \beta_1 & \beta_2 \end{array} \qquad \begin{array}{c} \alpha \\ \hline \alpha_1 \\ \alpha_2 \end{array}$$

Definição 3.6

Seja Δ um ramo do Tableau composto por um conjunto $\{\psi_1, \psi_2, \ldots, \psi_i, \ldots \psi_n\}$ de fórmulas. Uma expansão do ramo Δ é obtida pelo desenvolvimento de uma das fórmulas ψ_i pertencentes a Δ.

❐

[1] Na forma como é tradicionalmente apresentada, a eliminação da dupla negação $\sim\sim\omega$ / ω ω é classificada como uma regra do tipo α e provoca a criação de duas instâncias da fórmula ω duplamente negada. Claro está que a classificação da dupla negação como um regra do tipo α objetiva facilitar a formalização, evitando a criação de uma regra específica. Esse procedimento se justifica pelo fato de o desenvolvimento da dupla negação não provocar a bifurcação do ramo onde ela ocorre.Todavia, na prática, a regra utilizada é $\sim\sim\omega$ / ω onde apenas uma das instâncias é criada.

Se ψ_i é do tipo α, a expansão de $\Delta=\{\psi_1,\psi_2, \ldots,\alpha,\ldots \psi_n\}$ pelo desenvolvimento de α é o ramo $\Delta'=\{\psi_1,\psi_2, \ldots,\alpha,\ldots \psi_n,\alpha_1,\alpha_2\}$.

Se ψ_i é do tipo β, a expansão de $\Delta=\{\psi_1,\psi_2, \ldots,\beta,\ldots \psi_n\}$ pelo desenvolvimento de β provoca uma bifurcação de Δ em dois ramos: $\Delta_1'=\{\psi_1,\psi_2, \ldots,\beta,\ldots \psi_n,\beta_1\}$ e $\Delta_2'=\{\psi_1,\psi_2, \ldots,\beta,\ldots \psi_n,\beta_2\}$. Enfatizando apenas os ramos que estão sendo expandidos, e omitindo os demais ramos da árvore, teríamos, respectivamente:

$$\begin{array}{c} \psi_1 \\ \psi_2 \\ \vdots \\ \alpha \\ \vdots \\ \psi_n \\ \hline \alpha_1 \\ \alpha_2 \end{array} \qquad\qquad \begin{array}{c|c} \multicolumn{2}{c}{\psi_1} \\ \multicolumn{2}{c}{\psi_2} \\ \multicolumn{2}{c}{\vdots} \\ \multicolumn{2}{c}{\beta} \\ \multicolumn{2}{c}{\vdots} \\ \multicolumn{2}{c}{\psi_n} \\ \hline \beta_1 & \beta_2 \end{array}$$

Definição 3.7

Um Tableau $\mathcal{T}$ para uma fórmula ω é uma árvore tal que:

a) a raiz de $\mathcal{T} = \sim\omega$ é um tableau para ω.

b) Se $\mathcal{T}$ é um Tableau para ω e Δ um ramo de $\mathcal{T}$, então:

$\mathcal{T}'$, gerado pela expansão do ramo Δ, também é um Tableau para ω.

❐

Definição 3.8

Um ramo θ é fechado quando contiver dois literais complementares L e ~L.

❐

Cada ramo fechado é insatisfatível.

Definição 3.9

Um Tableau $\mathcal{T}$ é fechado se todos os ramos são fechados.

❑

Temos uma prova para a fórmula ψ, se o Tableau para ψ é fechado.

Exemplo 3.8

Exemplo de Tableau para $(a \to b) \wedge (b \to c) \to (a \to c)$

$$\sim(((a \to b) \wedge (b \to c)) \to (a \to c))$$
$$(a \to b) \wedge (a \to b))$$
$$\sim(a \to c)$$
$$(a \to b)$$
$$(a \to c)$$
$$a$$
$$\sim c$$

$$\begin{array}{c|c}
\sim a & b \\
\otimes & \begin{array}{c|c} \sim a & c \\ \otimes & \otimes \end{array}
\end{array}$$

◆

3.2.2 - Tableaux para Lógica de Primeira Ordem

Para o caso da lógica de predicados, além das regras dos tipos α e β, são acrescentadas as regras dos tipos γ e δ, a fim de desenvolver as fórmulas quantificadas universalmente e existencialmente.

Definição 3.10

Chamamos desenvolvimento de uma fórmula ψ à decomposição de ψ em suas subfórmulas, segundo as regras do tipo α, tipo β, bem como dos tipos γ e δ abaixo.

Regras Tipo γ

$$\frac{\forall X\ \gamma}{\begin{array}{c}\forall X\ \gamma \\ \gamma(X/Z)\end{array}} \qquad \frac{\sim \exists X\ \gamma}{\begin{array}{c}\sim \exists X\ \gamma \\ \sim \gamma(X/Z)\end{array}}$$

,onde z é uma variável nova.

Regras Tipo δ

$$\frac{\exists X\ \delta}{\delta(X/t)} \qquad \frac{\sim \forall X\ \delta}{\sim \delta(X/t)}$$

, onde t é um termo que se obtém por skolemização.

❐

Os conceitos de ramo fechado , Tableau fechado e prova são os mesmos da lógica proposicional.

Exemplo 3.9

Exemplo de Tableau para $(\forall X(p(X) \rightarrow q(b)) \rightarrow \exists Y(p(Y) \rightarrow q(b)))$

1)	$\sim(\forall X\,(p(X) \rightarrow q(b)) \rightarrow \exists Y\,(p(Y)\rightarrow q(b))\,)$			
2)	$\forall X\,(p(X) \rightarrow q(b))$			2 e 3: de 1 por α
3)	$\sim\exists Y\,(p(Y)\rightarrow q(b))$			
4)	$p(Z) \rightarrow q(b))$			4: de 2 por γ
5)	$\sim p(Z)$	9)	$q(b)$	5 e 9: de 4 por β
6)	$\sim(\,p(W)\rightarrow q(b)\,)$	10)	$\sim(\,p(K)\rightarrow q(b)\,)$	6 e 10: de 3 por γ
7)	$p(W)$	11)	$p(K)$	7 e 8: de 6 por α
8)	$\sim q(b)$	12)	$\sim q(b)$	fecha $\sim p(Z)$ com $p(W)$
	$\otimes$		$\otimes$	11e 12: de 10 por α
				fecha $q(b)$ com $\sim q(b)$

◆

O método de prova por Tableaux para lógica proposicional é completo, correto e, portanto, nesse contexto, é um algoritmo de decisão (Smullyam,1968).

Para lógica de primeira ordem, obviamente, o procedimento não é decidível. As regras do tipo γ permitem a inclusão de infinitas novas variáveis na prova. Assim sendo, é necessário estabelecer um nível arbitrário a partir do qual consideramos que o procedimento de prova falha.

A maioria dos autores (Fitting,1990) estabelece como nível um tamanho máximo para o ramo. Se qualquer ramo atinge o tamanho máximo arbitrado, então ocorre um backtracking. Em Mendes(1972), o nível é um limite no número de variáveis a serem empregadas na prova. Outros níveis podem ser empregados, como por exemplo:

a) o número máximo de vezes em que cada fórmula é utilizada na prova.

b) o número máximo de vezes em que cada fórmula é utilizada no ramo.

c) o número máximo de símbolos funcionais dentro de cada argumento associado com um limite no número de vezes que uma variável livre nova é utilizada num argumento.

Se utilizarmos como nível o critério a) e fixarmos o número máximo de vezes em que cada fórmula é usada na prova como igual a 1, temos um procedimento de prova para lógica linear (Carvalho,1995).

Outra questão a ser considerada são as heurísticas disponíveis para o método de Tableaux. A primeira heurística, denominada Ciranda de fórmulas (Carvalho,1995), corresponde a postergar a reutilização de fórmulas até que todas as outras possíveis sejam utilizadas. Fitting atribui a Smullyam a adoção dessa heurística para o método de Tableaux (Fitting,1990). A Ciranda é fundamental, tanto para o método de Tableaux como para a Resolução. Sem a sua utilização, qualquer dos dois métodos entra rapidamente em loop.

Outra heurística freqüentemente utilizada (Smullyan,1968), (Mendes,1972) consiste em priorizar o desenvolvimento das fórmulas tipo α e δ , em seguida às do tipo β e, por último, às do tipo γ. Vamos denominar essa heurística de heurística da menor bifurcação. Através de sua utilização, pretende-se primeiro desenvolver as fórmulas que não implicam a bifurcação de ramos, e, por último, as fórmulas quantificadas universalmente que podem gerar mais de uma instância de cada fórmula.

3.3 - Critérios Para Avaliação de Eficiência dos Métodos de Prova

Devemos entender o processo de prova automática como uma busca dentro do espaço das fórmulas que podem ser concluídas a partir de uma base de conhecimento. Como estamos tratando de provas por refutação, o objetivo do processo de busca é encontrar uma contradição (a cláusula vazia, no caso da Resolução, ou o fechamento de todos os ramos, no caso do método de Tableaux).

Os critérios abaixo relacionados para essa análise espelham esse enfoque. Estamos procurando verificar características presentes nos métodos analisados que venham a favorecer uma busca mais eficiente.

- **O desenvolvimento da prova** - O encaminhamento através do qual o método procura sistematicamente atacar todos os caminhos possíveis. Dentro do direcionamento adotado, cada método permite e induz a seleção de heurísticas para melhorar a escolha do próximo passo.
- **O reaproveitamento de lemas** - Em que medida o método reaproveita o conhecimento obtido na busca, através dos caminhos de prova anteriormente pesquisados.
- **A percepção da falha** - Como e em que condições o método conclui que o caminho que está sendo desenvolvido é infrutífero.
- **A recuperação da busca após a falha** - Como se dá o *backtracking*, e em que o conhecimento obtido durante a falha ajuda a redirecionar ou podar o restante da prova. Podemos entender a recuperação da busca após a falha como uma forma de reaproveitar o conhecimento obtido acerca dos lemas que falharam.

3.3.1 - O Desenvolvimento da Prova

O encaminhamento dentro do método de Tableaux corresponde a uma busca em largura. Esse encaminhamento confere ao Tableau um caráter sistemático que o torna de fácil entendimento e didaticamente apropriado para um primeiro contato com a prova automática. Para lógica proposicional não há geração de ciclos durante a prova. Para lógica de primeira ordem, os ciclos são possíveis apenas porque ao desmembrarmos as fórmulas quantificadas universalmente não as retiramos da base. Um ciclo infinito seria gerado num ramo do Tableau, se nos restringíssemos a desmembrar infinitamente sempre uma mesma fórmula ou um conjunto de fórmulas.

Claro está que uma pessoa, executando uma prova manualmente sobre o papel, não incorreria em tal ciclo. O método nos induz a desmembrar fórmulas sucessivamente distintas umas das outras. Todavia, quando consideramos a prova automatizada, devemos utilizar a Ciranda de Fórmulas, para que essa repetição não aconteça.

A principal desvantagem da busca em largura utilizada no Tableau é a complexidade que lhe é inerente (Mendes,1972). Tomemos como exemplo uma pequena base de conhecimento com 100 implicações. Teríamos até 2^{100} ramos. Em geral, para n alternativas teríamos até 2^n ramos, o que é absolutamente inaceitável. Quando pensamos em utilizar Tableaux numa base de conhecimento, mesmo pequena, verificamos que o método é impraticável se não houver um critério para escolha de quais dessas alternativas serão importantes para a prova.

Evidentemente, o problema da complexidade torna impossível supor que a prova de um goal G em relação a uma base de fórmulas B_1 , B_2 ... B_n possa ser reduzida à prova do teorema: $B_1 \wedge B_2 \wedge \ldots \wedge B_n \rightarrow G$ (apesar de a equivalência lógica estar correta). Provar um teorema contra uma base de conhecimento é, na prática, diferente de provar pequenas tautologias.

Alguns autores, embora ciosos e atentos aos diversos aspectos teóricos que envolvem a prova com Tableaux, esquecem em sua análise o problema de complexidade. Avron (1993), por exemplo, reconhece que "O método de Tableaux 'clássico' não é prático para qualquer coisa que supere o trivial". Todavia, esquecendo o problema da complexidade, afirma que "no contexto da lógica proposicional, o método de Tableaux é melhor do que o método da resolução".

Esta última afirmação só pode ser considerada válida para pequenas e "triviais" tautologias, cuja prova pode ser realizada ou acompanhada manualmente, sem muita dificuldade, e que, justamente por isso, são apropriadas a servir de exemplo em trabalhos científicos e compêndios didáticos.

A concepção de heurísticas para o método de Tableaux foi pouco estudada. Isso se deve em parte ao sucesso obtido pela Resolução e, em parte, porque o método foi concebido, inicialmente, como uma ferramenta gráfica, para ser usado por pessoas, analisando pequenas fórmulas, e não para prova automática. O método supõe, a cada passo, que a pessoa use a própria inteligência, com o intuito de conduzir a prova selecionando as fórmulas que serão decompostas.

As estratégias propostas para o método de Tableaux, via de regra, propõem que as fórmulas sejam desmembradas de acordo com o seu operador mais externo, segundo a seguinte ordem:

1) conjunções (regra α)

2) fórmulas quantificadas existencialmente (regra γ)

3) disjunções (regra β)

4) fórmulas quantificadas universalmente (regra δ)

A Resolução, de modo diverso, já foi criada tendo em vista a prova automática. Assim, desde cedo foram desenvolvidas heurísticas que lhe conferiam um desempenho aceitável.

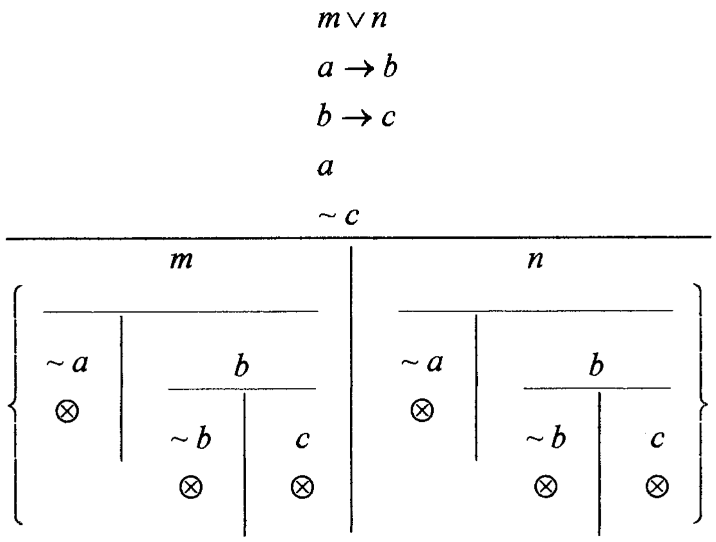
A estratégia do conjunto suporte associada à estratégia linear implicam um encaminhamento em profundidade direcionado para a pergunta. Essa associação de estratégias é a mais largamente utilizada na resolução. Na verdade, a busca em largura, que caracteriza a resolução por saturação, é utilizada apenas nas provas de completude e corretude do método.

Em (Costa,1992), vimos proposta a utilização da estratégia linear com o método de Tableaux.

3.3.2 - O Reaproveitamento de Lemas

No método de Tableaux, o reaproveitamento de lemas não existe. No caso proposicional, uma vez fechado um ramo, nenhuma informação é transmitida aos ramos seguintes. No caso da lógica de predicados, são transmitidas apenas as instanciações das variáveis. O exemplo abaixo ilustra a ausência do reaproveitamento de lemas no Método de Tableaux. Observa-se que o desenvolvimento dos dois sub-tableaux assinalados com chaves é absolutamente igual.

Exemplo 3.10



◆

Na Resolução, também não encontramos o reaproveitamento de lemas. A rigor, o método não armazena informações sobre os caminhos já executados. O exemplo abaixo, na Prova 1, ilustra a repetição de uma subprova: o literal "a" é resolvido na cláusula 7) e depois, novamente, na cláusula 11).

Todavia, dependendo da ordem na qual as cláusulas são provadas, podemos ter uma redução de dois literais semelhantes (Prova 2 do exemplo abaixo). Nesse caso, ocorre concomitantemente à eliminação de um dos ramos em duplicidade.

Exemplo 3.11

	1)	$\sim m \vee a$
	2)	$\sim n \vee a$
	3)	$\sim a \vee b$
	4)	$\sim b \vee c$
	5)	$\sim c$
	6)	$\sim c \vee a$
goal	7)	$m \vee n$

PROVA 1

de 7) e 1) =	8)	$\underline{a} \vee n$
de 8) e 3) =	9)	$b \vee n$
de 9) e 4) =	10)	$c \vee n$
de 10) e 5) =	11)	n
de 11) e 2) =	12)	$\underline{a}$
de 12) e 3) =	13)	b
de 13) e 4) =	14)	c
de 14) e 5) =	15)	$\otimes$

PROVA 2

de 7) e 1) =	8)	$a \vee n$
de 8) e 2) =	9)	$a \vee a$
REDUÇÃO	10)	a
de 10) e 3) =	11)	b
de 11) e 4) =	12)	c
de 12) e 5) =	13)	$\otimes$

PROVA 3

de 7) e 1) =	8)	$\underline{a \vee n}$
de 8) e 3) =	9)	$b \vee n$
de 9) e 4) =	10)	$c \vee n$
de 10) e 6) =	11)	$\underline{a \vee n}$
de 11) e 3) =	12)	$b \vee n$
de 12) e 4) =	13)	$c \vee n$
de 13) e 6) =	14)	$\underline{a \vee n}$

loop

●

Proposição: Na resolução sempre é possível encontrar um caminho de prova no qual a aplicação da regra de redução elimina a necessidade do reaproveitamento de lemas.

A questão é que, contando apenas com os recursos do método, não é possível saber de antemão se haverá ou não a repetição de um literal durante o caminho de prova. Portanto, não conduzimos a prova na direção de uma cláusula com literais iguais somente para ter a possibilidade de aplicar a regra de redução.

A rigor, o método não reaproveita lemas porque sequer armazena informações sobre os caminhos já executados. Na Prova 3, vemos, para o mesmo exemplo acima, um loop no qual a cláusula "a $\vee$ n" pode ser repetida infinitas vezes.

3.3.3 - A Percepção da Falha

Na Resolução a percepção da falha se dá quando encontramos um literal que não conseguimos unificar com nenhum outro. No caso proposicional, toda a cláusula é retirada da base, não sendo considerada nos demais passos da prova. No caso de primeira ordem, ocorre o *backtracking*, e uma nova instância das variáveis é testada.

No Tableau proposicional não existe a falha de um caminho, mas tão-somente a falha de toda a prova. A decomposição de fórmulas irrelevantes acarreta a multiplicação de caminhos que contêm basicamente a mesma subprova. No exemplo 3.11 a decomposição da fórmula "m $\vee$ n", que não é relevante para a prova, provoca a duplicação do número de ramos que deverão ser analisados.

No Tableau de primeira ordem só ocorre a falha num ramo quando a prova atinge o nível máximo especificado. Subentende-se, então, que uma das unificações realizadas durante o fechamento de ramos anteriores foi inadequada. O *backtracking* ocorre e os ramos anteriormente fechados são reabertos, à procura de outras instanciações para as variáveis que serão transmitidas adiante através da prova.

Claro está que, embora não identifique ciclos, nem reconheça literais que já falharam anteriormente, a percepção da falha em um ramo na Resolução é muito mais eficiente do que no Tableau.

3.3.4 - A Recuperação da Busca após a falha

Consideraremos nesta seção como acontece a recuperação da busca após a falha na Resolução em , em seguida, no método de Tableaux

3.3.4.1- Recuperação da busca após a falha – Resolução

A recuperação de falha na Resolução proposicional pode ser feita de forma extremamente satisfatória. Quando encontramos um literal, isolado e retiramos da base a cláusula na qual ele ocorre, evitamos a reincidência da mesma falha. Mais do que isso, evitamos que a mesma cláusula e o mesmo literal possam ser considerados através de outros caminhos.

Todavia, as duplicidades geradas no processo de transformação da fórmula para a forma clausal não são consideradas. Sejam, no exemplo, as cláusulas 1) e 2) geradas a partir da fórmula $(d \vee a) \wedge (b \vee d)$. Consideremos o trecho da prova apresentado para o objetivo ~d.

Exemplo 3.12

1) $a \vee b \ \vee d$
2) $a \vee c \ \vee d$
3) $\sim b \vee .... \vee ...$
4) $\sim c \vee .... \vee ...$

---

goal 5) $\sim d$

de 1) e 5) = 6) $a \vee b$ - falha por não resolver "a" retira a fórmula 6) da base.

◆

O procedimento deve fazer o *backtracking*, e, somente quando vier a considerar o literal "a" da cláusula 2), é que falhará novamente, eliminando a outra cláusula que foi gerada a partir da fórmula $(d \vee a) \wedge (b \vee d)$.

Na resolução em primeira ordem, a recuperação da falha não é tão fácil. Quando ocorre uma falha, recorre-se ao *backtracking*, e uma nova unificação para as variáveis é procurada. O *backtracking* é feito primeiramente para o último literal instanciado na cláusula, e não para a cláusula anterior, como no caso proposicional. Se as variáveis forem unificadas com os mesmos valores o erro se repete.

Além disso, mesmo que falhem todas as tentativas realizadas por um caminho, não é lícito retirar a cláusula da base, uma vez que, através de outro caminho, a mesma cláusula pode ser acessada com as variáveis instanciadas de forma completamente diversa.

3.3.4.2 - Recuperação da falha no método de Tableaux

No método de Tableaux, a percepção da falha só ocorre no cálculo de primeira ordem. E, mesmo assim, só quando a prova atinge o limite máximo estabelecido para desistência. Quando isso acontece, uma grande quantidade de ramos irrelevantes já deve ter sido gerada.

As unificações ocorrem no momento de fechamento de cada ramo. Portanto, a recuperação da falha consiste em fazer o backtracking até o último ramo que foi fechado, reabri-lo e tentar outra possibilidade de fechamento com novas instanciações.

3.5 - Comparação Tableaux X Resolução

Identificamos, em nossa comparação dos métodos de Tableaux e Resolução, as seguintes principais deficiências do método de Tableaux:

a) As estratégias utilizadas para o método de Tableaux valem-se de um encaminhamento em largura, aceitável para pequenos problemas, mas completamente inapropriado, se objetivarmos provar uma fórmula a partir de uma base de conhecimento. Neste último caso, em face da explosão combinatória, a prova deve, necessariamente, utilizar uma estratégia goal-oriented. Em (Costa,1992), encontramos um tentativa de contornar o problema através da aplicação da estratégia linear para o método de Tableaux.

b) A identificação da falha, no método de Tableaux, só ocorre quando é atingido o nível máximo de pesquisa. Em comparação, a falha na resolução acontece tão logo é encontrado um literal para o qual não dispomos de outro literal complementar contra quem resolver. Ao identificar mais cedo a falha, a resolução poda caminhos infrutíferos.

c) Uma vez que o Tableau não identifica a falha na ocorrência de literais isolados, a recuperação da falha, nesse caso, também não está definida. Para o método de Tableaux, não seria correto simplesmente fazer o *backtracking* e pesquisar outra fórmula, como é feito na Resolução. A ocorrência de um literal isolado numa cláusula nos habilita a descartá-la. A ocorrência de um literal isolado numa fórmula genérica nos habilita somente a simplificá-la.

4. MÉTODO DE SIMPLIFICAÇÃO POR ELIMINAÇÃO DE LITERAIS

Este capítulo apresenta o Método de Prova utilizando Simplificação por Eliminação de Literais (MSEL). Primeiramente é apresentado o caso proposicional. Em seguida, fazem-se ajustes análogos ao método de eliminação de modelos, de forma a garantir a completude do MSEL. Ao final é apresentada a versão para lógica de primeira ordem.

O MSEL pode ser aplicado a fórmulas quaisquer, todavia, como mostraremos no capítulo 5, se restringirmos sua aplicação à cláusulas, obteremos os mesmos passos de inferência que aqueles obtidos com o método de Resolução.

4.1 - MSEL PROPOSICIONAL

O MSEL é um método refutacional, que, da mesma forma que a Resolução, é aplicado a um par de fórmulas que possuem dois literais de polaridade complementar. Arbitrariamente, denominamos uma das fórmulas de hospedeira e a outra de visitante.

Seja L_h um literal da fórmula hospedeira, e seja L_v um literal da fórmula visitante, tais que L_h e L_v são complementares.

O método consiste em simplificar a fórmula visitante, eliminando o literal L_v, e, em seguida, substituir o literal L_h da fórmula hospedeira pelo resultado da simplificação da fórmula visitante. O diagrama da Figura 4.1 exemplifica o método.

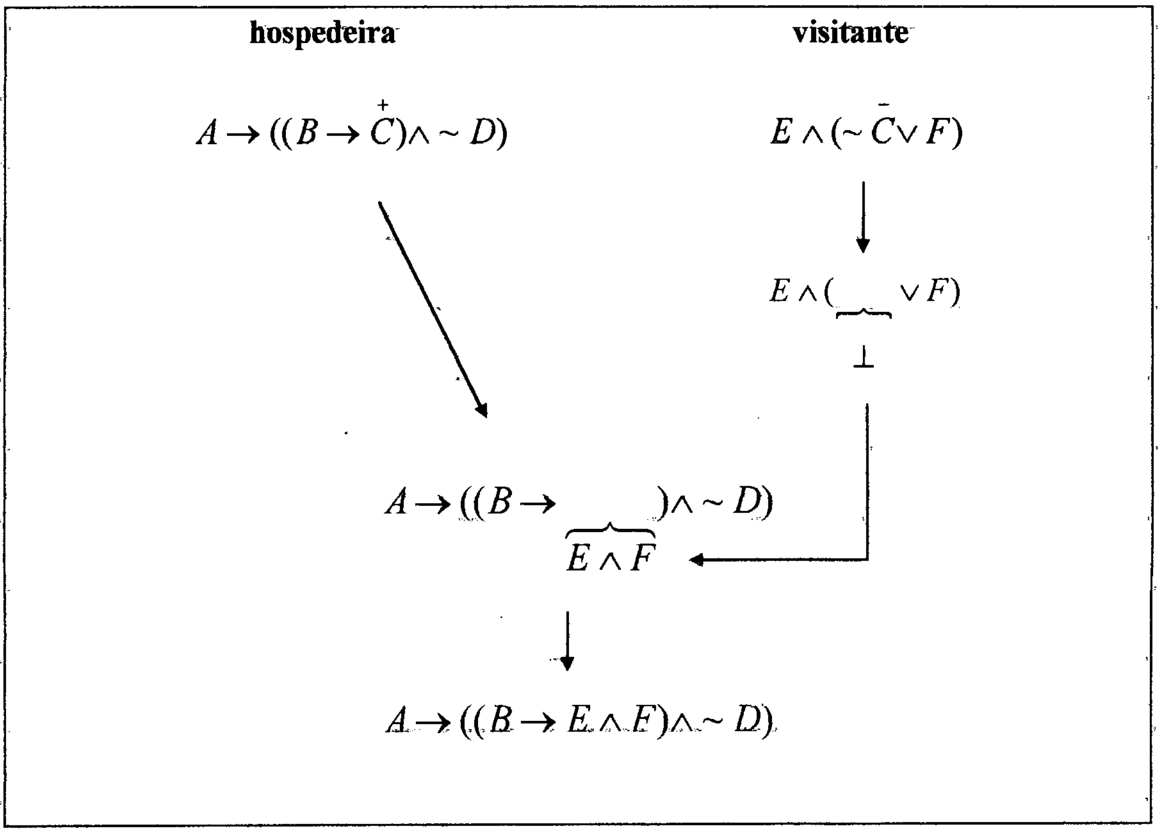


Figura 4.1 - Exemplo de Simplificação por Eliminação de Literais

A seguir demonstra-se como, a partir de duas fórmulas h e ν, obtêm-se, através do MSEL, uma fórmula resultante r, tal que $h, \nu \vdash r$.

De duas fórmulas

h (a meta - que será chamada de hospedeira) e

ν (uma fórmula da base - que será chamada visitante)

que possuem literais L_h e L_v complementares podemos deduzir uma fórmula resultante r, utilizando um procedimento de duas fases.

Fase 1)

1.1 - substituição com ajuste de polaridade de L_V por uma constante lógica ($\perp$) na fórmula ν (visitante):

$$\nu\,(Lv \,//\, \perp)$$

1.2 - simplificação da fórmula obtida na fase 1.1, utilizando-se as regras abaixo. Seja simplify a função correspondente à simplificação, obtemos ω como resultado da fase 1.

$$\omega = \text{simplify}\,(\, \nu\,(\, L_v \,//\, \perp\,)\,)$$

As simplificações objetivam eliminar a ocorrência de constantes lógicas. Se o resultado da simplificação é uma nova constante lógica, então, uma nova simplificação será feita. As simplificações continuarão até que sejam eliminadas as constantes lógicas inseridas em ω, ou até que ω seja reduzido a uma constante lógica, e nenhuma regra de simplificação possa ser utilizada.

Regras de Simplificação:

$$\text{F1)}\ \frac{(\alpha \vee \perp)}{\alpha} \qquad \text{F2)}\ \frac{\alpha \wedge \perp}{\perp} \qquad \text{F3)}\ \frac{(\perp \rightarrow \alpha)}{\mathrm{T}} \qquad \text{F4)}\ \frac{(\alpha \rightarrow \perp)}{\sim \alpha} \qquad \text{F5)}\ \frac{\sim \perp}{\mathrm{T}}$$

$$\text{V1)}\ \frac{(\alpha \vee \mathrm{T})}{\mathrm{T}} \qquad \text{V2)}\ \frac{\alpha \wedge \mathrm{T}}{\alpha} \qquad \text{V3)}\ \frac{(\mathrm{T} \rightarrow \alpha)}{\alpha} \qquad \text{V4)}\ \frac{(\alpha \rightarrow \mathrm{T})}{\mathrm{T}} \qquad \text{V5)}\ \frac{\sim \mathrm{T}}{\perp}$$

Fase 2)

2.1 - substituição com ajuste de polaridade de L_h por ω, na fórmula *h* (hospedeira):

$$h\,(\, L_h \,//\, \omega)$$

2.2 - a simplificação da fórmula obtida gerará a resultante *r*

$$r = \text{simplify}\,(\, h\,(\, Lh \,//\, \omega)\,)$$

Evidentemente, essa simplificação só será necessária quando ω for uma constante lógica. Também é claro que as simplificações das fases 1.2 e 2.2 podem ser resumidas numa só simplificação. Podemos reescrever, agrupando as fases 1 e 2:

$$h, \nu \vdash r = \text{simplify}\,(\, h\,(\,\text{Lh}\ //\ \nu\,(\,\text{Lv}\ //\ \perp\,)\,)\,).$$

Ou, ainda, tornando implícitas as simplificações, teríamos a resultante r como resultados de duas substituições com ajuste de polaridade:

$$h, \nu \vdash r = h\,(\,L_h\ //\ \nu\,(\,L_v\ //\ \perp\,)\,)$$

Definição 4.1

Sejam h e v fórmulas que possuem, respectivamente, literais L_h e L_v complementares. A resultante r de h e v é a fórmula: $h\,(\,L_h\ //\ \nu\,(\,L_v\ //\ \perp\,)\,)$

❑

Veja, a seguir, um exemplo (Modus Ponens) detalhando as duas fases da dedução, segundo o MSEL:

Exemplo 4.1

Sejam h e ν as fórmulas indicadas abaixo:

$$\begin{array}{lll} h) & \overset{-}{A} \rightarrow \overset{+}{B} & \qquad\qquad L_h = \overset{-}{A} \\ \nu) & \overset{+}{A} & \qquad\qquad L_v = \overset{+}{A} \\ \hline r) & \overset{+}{B} & \end{array}$$

Indicando as substituições com ajuste de polaridade realizada, temos:

$$h)\quad \underbrace{\overset{-}{A}}_{\nu)\ \underbrace{\overset{+}{A}}_{\perp}} \quad \rightarrow B$$

Mostrando, passo a passo, a dedução, temos :

fase 1.1) substituição com ajuste de polaridade em ν: ν (L_V // ⊥)

$$\nu\,(L_v\ //\perp) = \nu\,(\overset{+}{A}\ //\perp)$$

Como pol($\overset{+}{A}$) = sinal($\overset{+}{A}$), não há necessidade de reajustar o sinal. Logo:

$$\nu\,(\overset{+}{A}\ //\perp) = \nu\,(\overset{+}{A}\ /\perp)$$

Efetuando a substituição, obtemos:

$$\nu\,(\overset{+}{A}\ /\perp) = \overset{+}{A}\ (\overset{+}{A}\ /\perp) = \perp$$

fase 1.2) simplificação de ν (L_V // ⊥)

$$\omega = \text{simplify}\,(\perp) = \perp$$

Indicando as substituições com ajuste de polaridade realizada, temos:

$$h)\quad \underbrace{\overset{-}{A}}_{\nu)\ \perp} \quad \rightarrow B$$

fase 2.1) substituição com ajuste de polaridade em *h*: *h* (L_h // ω)

$$h\,(L_h\ //\omega) = h\,(\overset{-}{A}\ //\omega)$$

Como pol($\overset{-}{A}$) = - e sinal($\overset{-}{A}$) = + , há necessidade de ajuste no sinal. Logo:

$$h\,(\overset{-}{A}\ //\omega) = h\,(\overset{-}{A}\ /{\sim}\omega) = h\,(\overset{-}{A}\ /{\sim}\perp) = h\,(\overset{-}{A}\ /\top)$$

Efetuando a substituição, obtemos:

$$\overset{-}{A} \rightarrow \overset{+}{B}\ (\overset{-}{A}\ /\top) = V \rightarrow \overset{+}{B}$$

fase 2.2) simplificação de *h* (L_h // ω)

$$r = \text{simplify}\,(h\,(\,L_h\ //\ \omega)\,) = \text{simplify}\,(\,\top \rightarrow \overset{+}{B}\,)\ =\ \overset{+}{B}$$

◆

Exemplo 4.2

Sejam *h* e ωas fórmulas indicadas abaixo:

$$\begin{array}{lll} h) & (\overset{+}{A} \vee \overset{+}{B}) \wedge \overset{+}{D} & L_1 = \overset{+}{B} \\ \nu) & (\overset{+}{A} \rightarrow \overset{-}{B}) \rightarrow \overset{+}{C} & L_2 = \overset{-}{B} \\ \hline r) & (\overset{+}{A} \vee \overset{+}{C}) \wedge \overset{+}{D} & \end{array}$$

Indicando as substituições com ajuste de polaridade realizada, temos:

$$h)\ A \vee \underbrace{\overset{+}{B}}_{\nu)\ (A \rightarrow \underbrace{\overset{-}{B}}_{\perp})\ \rightarrow C} \wedge D$$

Mostrando, passo a passo, a dedução, temos:

fase 1.1) substituição com ajuste de polaridade em ν: ν (L_ν // ⊥)

$$\nu\,(\,L\nu\ //\perp) = \nu\,(\overset{-}{B}\ \ //\perp)$$

Como pol($\overset{-}{B}$) = - e sinal($\overset{-}{B}$)= +, há necessidade de reajustar o sinal. Logo:

$$\nu\,(\overset{-}{B} // \perp) = \nu\,(\overset{-}{B}\ \ / \sim\!\perp) = \nu\,(\overset{-}{B}\ \ /\ T)$$

Efetuando a substituição, obtemos:

$$\nu\,(\overset{-}{B}\ \ /\top)\ =\ (\overset{+}{A} \rightarrow \overset{-}{B}) \rightarrow \overset{+}{C}\ \ (\overset{-}{B}\ \ /\top)\ =\ (\overset{+}{A} \rightarrow V) \rightarrow \overset{+}{C}$$

fase 1.2) simplificação de v (L_v // $\perp$)

$$\omega = \text{simplify}\,((\overset{+}{A} \to V) \to \overset{+}{C})) = V \to \overset{+}{C} = \overset{+}{C}$$

fase 2.1) substituição com ajuste de polaridade em *h*: *h* (L_h // ω)

$$h(L_h \;//\; \omega) = h(\overset{+}{B} \;//\; \omega)$$

Como pol($\overset{+}{B}$) = sinal($\overset{+}{B}$), não há necessidade de ajuste no sinal. Logo:

$$h(\overset{+}{B} // \omega) = h(\overset{+}{B} \;/\; \omega) = h(\overset{+}{B} \;/\; \overset{+}{C})$$

Efetuando a substituição, obtemos:

$$(\overset{+}{A} \vee \overset{+}{B}) \wedge \overset{+}{D} \quad (\overset{+}{B} \,/\, \overset{+}{C}) = (\overset{+}{A} \vee \overset{+}{C}) \wedge \overset{+}{D}$$

fase 2.2) simplificação de *h* (L_h // ω)

$$r = \text{simplify}\,(h(L_h \;//\; \omega)) = \text{simplify}\,((\overset{+}{A} \vee \overset{+}{C}) \wedge \overset{+}{D}) = (\overset{+}{A} \vee \overset{+}{C}) \wedge \overset{+}{D}$$

◆

Vejamos agora mais alguns exemplos de pequenas provas utilizando o MSEL. Nas provas mais complexas, utilizaremos as provas mais simples, sem detalhá-las, conforme exemplo 4.2. Desta forma, sem detalhes desnecessários, as provas mais complexas ficarão mais legíveis.

Exemplo 4.3

A→ B, ~B ⊢ ~A **(Modus Tollens)**

$$A \to \underbrace{\overset{+}{B}}_{\underbrace{\sim\overset{-}{B}}_{\perp}} \Rightarrow A \to \underbrace{\overset{+}{B}}_{\perp} \Rightarrow A \to \perp \Rightarrow \sim A$$

Exemplo 4.4

A → B, B→ C ⊢ A→ C **(Silogismo Hipotético)**

$$A \to \underbrace{\overset{+}{B}}_{\underbrace{\overset{-}{B}}_{\perp} \to C} \Rightarrow A \to \underbrace{\overset{+}{B}}_{\mathrm{T} \to \mathrm{C}} \Rightarrow A \to \underbrace{\overset{+}{B}}_{C} \Rightarrow A \to C$$

Exemplo 4.5

A∨B , ~A ⊢ B **(Silogismo Disjuntivo)**

$$\underbrace{\overset{+}{A}}_{\underbrace{\sim\overset{-}{A}}_{\perp}} \vee B \Rightarrow \underbrace{\overset{+}{A}}_{\perp} \vee B \Rightarrow \perp \vee B \Rightarrow B$$

Exemplo 4.6

A∨B, A → C , B→D ⊢ C ∨ D **(Dilema Construtivo)**

$$\underbrace{\overset{+}{A}}_{\underbrace{\overset{-}{A}}_{\perp} \to C} \vee \underbrace{\overset{+}{B}}_{\underbrace{\overset{-}{B}}_{\perp} \to D} \Rightarrow \underbrace{\overset{+}{A}}_{\mathrm{T} \to C} \vee \underbrace{\overset{+}{B}}_{\mathrm{T} \to D} \Rightarrow \underbrace{A}_{C} \vee \underbrace{B}_{\mathrm{D}} \Rightarrow C \vee D$$

Exemplo 4.7

~B ∨ ~D, (A →B) , (C→D) ⊢ ~A ∨ ~C   **(Dilema Destrutivo)**

$$\begin{matrix} \sim\bar{B} & \vee & \sim\bar{D} \\ \overbrace{A\to \overset{+}{B}} & & \overbrace{C\to \overset{+}{D}} \\ \overbrace{\perp} & & \overbrace{\perp} \end{matrix} \Rightarrow \begin{matrix} \sim B & \vee & \sim D \\ \overbrace{A\to\perp} & & \overbrace{C\to\perp} \end{matrix} \Rightarrow \begin{matrix} \sim B & \vee & \sim D \\ \overbrace{\sim A} & & \overbrace{\sim C} \end{matrix} \Rightarrow \sim A \vee \sim C$$

◆

Exemplo 4.8

A→ B, C→ ~B ⊢ A →~C

$$\begin{matrix} A\to \overset{+}{B} \\ \overbrace{C\to \sim\bar{B}} \\ \overbrace{\perp} \end{matrix} \Rightarrow \begin{matrix} A\to \overset{+}{B} \\ \overbrace{C\to\perp} \end{matrix} \Rightarrow \begin{matrix} A\to \overset{+}{B} \\ \overbrace{\sim C} \end{matrix} \Rightarrow A \to \sim C$$

◆

Exemplo 4.9

A→ B, ~A→C ⊢ ~C → B

$$\begin{matrix} \bar{A} \to B \\ \overbrace{\sim \overset{+}{\bar{A}} \to C} \\ \overbrace{\perp} \end{matrix} \Rightarrow \begin{matrix} \bar{A} \to B \\ \overbrace{\mathrm{T} \to \mathrm{C}} \end{matrix} \Rightarrow \begin{matrix} \bar{A} \to B \\ \overbrace{C} \end{matrix} \Rightarrow \sim C \to B$$

◆

No MSEL, a escolha da fórmula que será hospedeira e daquela que será visitante não é comutativa. Quando os resultados obtidos não são logicamente equivalentes, um deles subsume o outro. Definiremos posteriormente uma heurística apropriada para selecionar a fórmula que será a hospedeira e aquela que será a visitante.

No exemplo abaixo, se tomamos A $\vee \overset{+}{B}$ como hospedeira e C $\rightarrow$ (D $\wedge \sim \overset{-}{B}$) como visitante, deduzimos A $\vee$ ~C. Todavia, se, ao contrário, como no exemplo seguinte, tomamos C $\rightarrow$ (D $\wedge \sim \overset{-}{B}$) como hospedeira e A $\vee \overset{+}{B}$ como visitante, obtemos C $\rightarrow$ (D $\wedge$ A). Comparando os resultantes dos dois exemplos, vemos que C $\rightarrow$ (D $\wedge$ A) subsume A $\vee$ ~C.

Exemplo 4.10

h) $\quad A \vee \overset{+}{B} \qquad\qquad L_h = \overset{+}{B}$

ν) $\quad C \rightarrow (D \wedge \sim \overset{-}{B}) \qquad\qquad L_v = \sim \overset{-}{B}$

Fase 1) ω= simplify (C $\rightarrow$ (D $\wedge$ ~B) ($\sim \overset{-}{B}$ // $\perp$)) = simplify (C $\rightarrow$ (D $\wedge$ $\perp$)) = ~C
Fase 2) *r* = simplify (A $\vee$ B ($\overset{+}{B}$ // ~C)) = simplify (A $\vee$ ~C) = A $\vee$ ~C

◆

Exemplo 4.11

h) $\quad C \rightarrow (D \wedge \sim \overset{-}{B}) \qquad\qquad L_h = \sim \overset{-}{B}$

ν) $\quad A \vee \overset{+}{B} \qquad\qquad L_v = \overset{+}{B}$

Fase 1) ω= simplify (A $\vee$ B ($\overset{+}{B}$ // $\perp$)) = simplify (A $\vee$ $\perp$)) = A
Fase 2) *r* = simplify (C $\rightarrow$ (D $\wedge$ ~B) ($\sim \overset{-}{B}$ // A) = simplify (C $\rightarrow$ (D $\wedge$ A))
= C $\rightarrow$ (D$\wedge$A)

$$h : C \rightarrow (D \wedge \underbrace{\overset{-}{B}}_{v\,:\,A \vee \overset{+}{B}}) \quad \vdash \quad C \rightarrow (D \wedge A)$$

◆

O MSEL é um método refutacional baseado no fato de que, numa fórmula contraditória, existem necessariamente, dois literais complementares.

A fórmula é contraditória se, após uma conveniente eliminação de todos os pares de literais complementares, encontramos a fórmula vazia.

A questão reside justamente em descobrir quais são essas eliminações convenientes. A forma escolhida pelo usuário para representar as fórmulas fornece informações importantes para direcionar essa busca.

Consideremos o caso de toda base estar representada numa única fórmula, α. Cumpre notar que, normalmente, tal caso só ocorre em pequenos exemplos. Somente a propósito alguém escreveria uma fórmula com mais que algumas linhas. Outra possibilidade é a de uma fórmula contraditória α ser gerada no final do decurso de uma prova.

Para verificar se uma fórmula α é contraditória utilizando o MSEL, basta considerar α igual à fórmula hospedeira e à fórmula visitante ao mesmo tempo.

Exemplo 4.12

Sejam h e v a mesma fórmula indicada abaixo:

$$h)\quad \sim(\overset{+}{B}\to(\ (\overset{+}{C}\to(\overset{+}{D}\to \overset{-}{B}))\wedge \overset{-}{A}\ ))\qquad\qquad L_h=\overset{-}{B}$$

$$v)\quad \sim(\overset{+}{B}\to(\ (\overset{+}{C}\to(\overset{+}{D}\to \overset{-}{B}))\wedge \overset{-}{A}\ ))\qquad\qquad L_v=\overset{+}{B}$$

$$r)\quad \sim(\overset{+}{B}\to\ \overset{-}{A}\ )$$

Indicando as substituições com ajuste de polaridade realizada, temos:

$$h)\ \sim(B\to((C\to(D\to \underbrace{\overset{-}{B}}_{v)\ \sim(\underbrace{\overset{+}{B}}_{\perp}\ \to((C\to(D\to B))\wedge A))}))\wedge A))$$

Mostrando, passo a passo, a dedução, temos:

fase 1.1) substituição com ajuste de polaridade em ν: ν (L_v // $\perp$)

$$\nu\,(Lv\ //\perp) = \nu\,(\overset{+}{B}\ //\perp)$$

Como pol($\overset{+}{B}$) = sinal($\overset{+}{B}$), não há necessidade de reajustar o sinal. Logo:

$$\nu\,(\overset{+}{B}//\perp) = \nu\,(\overset{+}{B}\ /\perp)$$

Efetuando a substituição, obtemos:

$$\nu\,(\overset{+}{B}\ /\perp) = \ \sim(\overset{+}{B}\rightarrow(\ (\overset{+}{C}\rightarrow(\overset{+}{D}\rightarrow\overset{-}{B}))\wedge\overset{-}{A}\ ))\ \ (\overset{+}{B}/\perp) =$$

$$= \sim(\perp\rightarrow(\ (\overset{+}{C}\rightarrow(\overset{+}{D}\rightarrow\overset{-}{B}))\wedge\overset{-}{A}\ ))$$

fase 1.2) simplificação de ν (L_v // $\perp$)

$$\omega = \text{simplify}\,(\sim(\perp\rightarrow(\ (\overset{+}{C}\rightarrow(\overset{+}{D}\rightarrow\overset{-}{B}))\wedge\overset{-}{A}\ )))) \ = \ \sim(\top) = \perp$$

fase 2.1) substituição com ajuste de polaridade em h: h (L_h // ω)

$$h\,(L_h\ //\,\omega) = h\,(\overset{-}{B}\ //\,\omega)$$

como pol($\overset{-}{B}$) = - e sinal($\overset{-}{B}$) = + , há necessidade de ajuste no sinal, logo:

$$h\,(\overset{-}{B}//\,\omega) \ = \ h\,(\overset{-}{B}\ /\sim\omega) \ = \ h\,(\overset{-}{B}\ /\ \sim\perp) \ = h\,(\overset{-}{B}\ /\ \top)$$

efetuando a substituição obtemos:

$$\sim(\overset{+}{B}\rightarrow(\ (\overset{+}{C}\rightarrow(\overset{+}{D}\rightarrow\overset{-}{B}))\wedge\overset{-}{A}\ ))\ \ (\overset{-}{B}\ /\ \top) \ =$$

$$\sim(\overset{+}{B}\rightarrow(\ (\overset{+}{C}\rightarrow(\overset{+}{D}\rightarrow\top))\wedge\overset{-}{A}\ ))$$

fase 2.2) simplificação de *h* (Lh // ω)

$$r = \text{simplify}\,(h\,(\text{Lh}\ //\ \omega)\,) = \text{simplify}\,(\sim (\overset{+}{B} \rightarrow (\ (\overset{+}{C} \rightarrow (\overset{+}{D} \rightarrow \mathrm{T})) \wedge \overset{-}{A}\ ))) =$$

$$r = \sim (\overset{+}{B} \rightarrow (\ (\overset{+}{C} \rightarrow \mathrm{T}) \wedge \overset{-}{A}\ )) =$$

$$r = \sim (\overset{+}{B} \rightarrow (\ \mathrm{T} \wedge \overset{-}{A}\ )) =$$

$$r = \sim (\overset{+}{B} \rightarrow\ \overset{-}{A}\ )$$

◆

A condição, para que a fórmula seja contraditória, de existirem dois literais complementares, é necessária, mas não suficiente. Por exemplo, a tautologia $B \vee \sim B$ possui dois literais complementares, e não é contraditória, visto que seus literais estão separados por uma disjunção. Não é desejável aplicar o MSEL nesses casos.

Por uma questão de eficiência, define-se o filtro abaixo:

Definição 4.2

Seja ω uma fórmula com dois literais complementares $\overset{+}{L}$ e $\overset{-}{L}$. Seja $\eta = \mathcal{N}(\omega)$. Seja α uma subfórmula de η tal que $\overset{+}{L} \in \alpha$ e $\overset{-}{L} \notin \alpha$. Seja β uma subfórmula de η tal que $\overset{+}{L} \notin \beta$ e $\overset{-}{L} \in \beta$.

Aplica-se o MSEL se $\alpha \wedge \beta$ é subfórmula de η.

❐

Não é desejável aplicar o MSEL, quando $\alpha \vee \beta$ é subfórmula de η.

4.2 - Pré-Processamento das Fórmulas

Para de reduzir o tempo de prova, podemos efetuar as seguintes simplificações para cada uma das fórmulas constantes da base de conhecimento.

a) eliminar todos os literais complementares

b) marcar como não elimináveis os literais complementares que não atendem ao filtro acima.

c) transformar as fórmulas conjuntivas do tipos $\alpha \wedge \beta$ em duas fórmulas distintas α e β , respectivamente, transformar $\sim(\alpha \vee \beta)$ para $\sim\alpha$ e $\sim\beta$ e transformar $\sim(\alpha \rightarrow \beta)$ para α e $\sim\beta$.

Os mesmos procedimentos devem ser utilizados a cada nova fórmula a ser inserida na base de conhecimento durante a prova. Evidentemente, se alguma fórmula contraditória for encontrada, o resultado da eliminação de seus literais será a constante lógica $\perp$, pondo fim a prova.

4.3 - Estratégias de Prova utilizando o MSEL

O MSEL procura capturar a intenção expressa na forma como o usuário escreveu as fórmulas. As escolhas sintáticas do usuário desempenham um papel fundamental no desenvolvimento da prova. Essa característica básica, que faz o MSEL diferir de outros métodos, possibilita o desenvolvimento de novas heurísticas, além daquelas usualmente empregadas nos métodos baseados em resolução.

Por exemplo: o usuário tem a opção de escrever $B \rightarrow (C \wedge D)$ ou distribuir a implicação escrevendo duas fórmulas $B \rightarrow C$ e $B \rightarrow D$.

No primeiro dos casos, presume-se que C e D estão relacionados semanticamente e devem ser conduzidos em conjunto no decorrer da prova. Na presença de $A \rightarrow B$ e $B \rightarrow (C \wedge D)$ deduziremos $A \rightarrow (C \wedge D)$.

No segundo caso, teremos duas linhas de prova distintas. Na presença de $A \rightarrow B$, com $B \rightarrow C$, deduziremos $A \rightarrow C$, e, por outra linha de prova, com $B \rightarrow D$, deduziremos $A \rightarrow D$.

Veja a seguir, duas heurísticas que introduzem uma prioridade na seleção dos literais, em função da sintaxe utilizada pelo usuário. A adoção dessas estratégias não conflita com aquelas tradicionalmente usadas, que também podem e devem ser usadas.

Em particular, as estratégias de conjunto suporte e a ciranda de fórmulas devem, necessariamente, ser empregadas antes de todas as outras. A estratégia de conjunto suporte deve ser empregada a fim de orientar a prova em função da pergunta feita pelo usuário enquanto a ciranda de fórmulas, que evita selecionar às fórmulas as mais recentemente usadas, a fim de diminuir significativamente a ocorrência de loops nas provas.

4.3.1 - Escolha de literais com menor profundidade

Na escolha da fórmula que será a hospedeira e daquela que será a visitante, utilizamos o critério de escolher para hospedeira aquela cujo literal a ser substituído tem menor profundidade.

Da mesma forma, ao pesquisar a base para selecionar um literal complementar, preferimos aqueles de menor profundidade. Assim, o literal C da fórmula $A \wedge C$, cuja profundidade é 0, teria preferência sobre o literal C de $B \rightarrow C$, cuja profundidade é 1.

A adoção do critério dos literais de menor profundidade faz com que privilegiemos os fatos em relação às regras.

Um segundo critério é o de privilegiar fórmulas com menor profundidade. Assim, entre o literal A da fórmula $A \rightarrow B$, que tem profundidade 1, e o literal A da fórmula $A \rightarrow (D \wedge (C \rightarrow D))$, que tem profundidade 2, preferimos aquele pertencente à fórmula de menor profundidade.

Este segundo critério, na prática, é pouco efetivo. Observar a profundidade dos literais é suficiente para distinguir entre fatos e regras. De resto, a quase totalidade das fórmulas presentes numa base de conhecimento têm nível 1.

Para não aumentar a complexidade do programa, decidimos não implementar o segundo critério.

4.3.2 -Prioridade da posição dos literais em função da implicação

Se estamos tentando provar $A \rightarrow \overset{+}{B}$, e dispomos na base de conhecimento das fórmulas $D \rightarrow \sim\bar{B}$ e $\bar{B} \rightarrow C$, parece-nos respeitar a intenção do usuário tentar primeiramente o caminho que segue com $\bar{B} \rightarrow C$.

Se o literal B é o conseqüente de uma implicação, é razoável priorizar as fórmulas onde ele apareça como antecedente. Formaríamos uma cadeia progressiva de conseqüências do tipo: $A \rightarrow B$, $B \rightarrow C \vdash A \rightarrow C$

Por outro lado, se estamos tentando provar A→ B, e temos na base D → A e ~A→ G, parece razoável tentar primeiro o caminho que segue com D→ A, onde A é o conseqüente. Teríamos assim uma cadeia regressiva de conseqüências.

Quando o literal analisado é antecedente da implicação, priorizamos fórmulas onde seu complementar aparece como conseqüente.

De uma maneira geral procuramos adotar o encadeamento regressivo baseado na sintaxe escolhida pelo usuário para as implicações. Assim, se desejamos provar ~B, e temos na base as fórmulas A→B e ~B→C, vamos continuar a prova com A→B, onde B aparece como conseqüente.

Finalmente, preferimos fórmulas onde o literal aparece numa implicação àquelas em que aparece numa disjunção ou conjunção.

Gabbay nos apresenta em (Gabbay,1991), (Broda,1985) um provador goal-oriented. As heurísticas acima permitem ao provador atingir objetivo semelhante, funcionando goal-oriented sempre que possível.

4.4 - Msel e Eliminação de Modelos

Como sabemos, para preservar a completude na Resolução, temos de utilizar ou a fatoração dos literais ou a Eliminação de Modelos (Loveland,1978). Muitos autores já apresentam o método da Resolução com a fatoração embutida na definição (Chang,1973), ou, até mesmo, em textos mais abrangentes, a omitem em benefício da simplicidade da exposição (Rich,1994), (Patterson,1990), (Winston,1987).

As duas bases abaixo exemplificam a necessidade de fatoração ou de Eliminação de Modelos na Resolução:

Exemplo 4.13:

1)	$A \vee A$		
2)	$\sim A \vee \sim A$		
3)	$A \vee \sim A$	de 1) e 2)	: tautologia
4)	$\sim A \vee \sim A$	de 3) e 2)	: loop
5)	$A \vee A$	de 3) e 1)	: loop

◆

Exemplo 4.14:

1) $p(a) \vee p(X)$

2) $\sim p(a) \vee \sim p(X)$

◆

O exemplo 4.13 demonstra que, mesmo na lógica proposicional, a completude depende da fatoração. No caso, os literais repetidos da fórmula $A \vee A$ e da fórmula $\sim A \vee \sim A$ devem ser fatorados, respectivamente, para A e para ~A. Só então é possível encerrar a prova.

Uma alternativa é a utilização do método de eliminação de modelos, que conserva devidamente marcados os literais já resolvidos, permitindo uma posterior redução, caso haja repetição do literal complementar.

O exemplo 4.14 (Genesereth,1987), (Casanova,1987) mostra que, no caso da primeira ordem, não se trata da simples eliminação de literais repetidos. Devemos procurar unificar aqueles que possuem o mesmo predicado e o mesmo sinal, a fim de fatorá-los. Atenção especial deve ser tomada quando mais de uma fatoração é possível. Nesse caso deve-se incluir um ponto de backtracking no caminho da prova.

No MSEL, que, restrito a cláusulas, funciona com os mesmos passos da resolução, também devemos escolher entre as duas opções de implementação, fatoração ou eliminação de modelos.

Por uma questão de simplicidade de implementação, optamos pela eliminação de modelos. Assim, passamos a marcar como resolvidos os literais da fórmula hospedeira que já foram utilizados. Vamos marcar os literais já utilizados escrevendo-os entre colchetes.

A regra de inferência do MSEL com eliminação de modelos seria :

$$h, v \vdash r = h(L_h // [L_h] \wedge v(L_v // \perp))$$

ou seja , em vez de excluir L_h, passamos a apenas marcá-lo de forma a registrar que já o utilizamos em um passo da dedução. A regra acima corresponde à expansão na Eliminação de modelos.

Definição 4.3:

Se h , v são fórmulas que possuem, respectivamente, os literais de polaridade complementar L_h e L_v, então, a fórmula $r = h(L_h // [L_h] \wedge v(L_v // \perp))$ é uma expansão de h e v.

❑

A redução é obtida quando, após sucessivos passos, todos os literais da subfórmula visitante forem eliminados. Quando, após as devidas simplificações, temos que a subfórmula $v(L_v // \perp) = \perp$, então eliminamos o literal marcado automaticamente, em função do processo de simplificação, sem a necessidade de criarmos uma regra específica para tratar a redução uma vez que:

$$[L_h] \wedge v (L_v // \perp)) \quad = \quad [L_h] \wedge \perp \quad = \quad \perp$$

***Exemplo 4.15*:**

1)	$A \rightarrow B$	
2)	$B \rightarrow C$	
3)	$C \rightarrow D$	
4)	A	
5)	$\sim D$	
6)	$A \rightarrow [B] \wedge C$	Expansão de 1) e 2)
7)	$A \rightarrow [B] \wedge [C] \wedge D$	Expansão de 6) e 3)
8)	$A \rightarrow [B] \wedge [C] \wedge [D] \wedge \perp$	Expansão de 7) e 5)
8')	$A \rightarrow [B] \wedge [C] \wedge \perp$	Redução de 8)
8'')	$A \rightarrow [B] \wedge \perp$	
8''')	$\sim A$	
9)	$[\sim A] \wedge \perp$	Expansão de 8''') e 4)
9')	$\perp$	Redução

◆

Podemos utilizar uma contração, definida em seguida, quando o literal marcado pode ser simplificado com outro literal presente numa subfórmula visitante.

***Definição 4.4*:**

Se $[L_h] \wedge v$ é uma subfórmula de h, e L_v é um literal de v complementar a L_h, então obtemos uma contração de h ,substituindo L_v por $\perp$, com ajuste de polaridade.

$$h \vdash h(([L_h] \wedge v) \;/\; ([L_h] \wedge v (L_v // \perp)) \;)$$

❐

Embora seja mais eficiente, não é necessária a criação de uma regra específica para contração, como aquela apresentada na definição 4.4. É correto apenas simplificar o literal marcado $[L_h]$, contra outro literal complementar L_v de uma subfórmula visitante.

Preferimos a opção de não criar uma regra específica para contração, para simplificar as provas de completude e corretude.

Assim sendo, a única modificação necessária para utilização da Eliminação de Modelos com MSEL referiu-se à inclusão do literal marcado na regra de Expansão. Não foram criadas regras específicas, nem para Redução, nem para Contração.

O exemplo 4.16 mostra como ocorre a eliminação de modelos no contexto do MSEL.

***Exemplo 4.16*:**

1)	B	
2)	$B \rightarrow C$	
3)	$C \rightarrow \sim B \wedge D$	
4)	$[B] \wedge (C)$	Expansão de 1) e 2)
5)	$[B] \wedge ([C] \wedge (\sim B \wedge D))$	Expansão de 5) e 3)
6)	$[B] \wedge ([C] \wedge (\perp \wedge D))$	Contração de [b] e ~b utilizando a fórmula 5) concomitantemente como hospedeira e visitante
7)	$[B] \wedge ([C] \wedge \perp)$	Redução
8)	$[B] \wedge \perp$	Redução
9)	$\perp$	

◆

4.5 - Backtracking

Ao utilizarmos um mecanismo de busca em profundidade em qualquer método de prova, devemos fazer backtracking tão logo detecte-se que o caminho que está sendo percorrido é infrutífero.

Segundo os critérios que estabelecemos no Capítulo 2, as questões relevantes são: como identificar que o caminho é infrutífero e como conduzir a prova após a falha.

Apesar de fundamentais, quando se considera o desempenho de um método, essas questões não são relevantes para a prova de completude. Enquanto para a prova de completude nos interessa demonstrar que sempre existe uma refutação, ao analisarmos os mecanismos de busca estamos interessados em encontrar mais eficientemente essa solução.

Como vimos no Capítulo 2, a resolução detecta a ocorrência de uma falha logo que encontra um literal que não pode ser resolvido. Por seu turno, o método de Tableaux faz backtracking apenas no cálculo de primeira ordem e através de um limite no tamanho da prova, que provoca o retrocesso, caso seja superado.

Consideramos a detecção de falha da Resolução mais eficiente que a do método de Tableaux. Assim sendo, respondendo à primeira questão, estabelecemos para o MSEL o mesmo critério de detecção de erros que na Resolução.

Definimos um ponto de backtracking para cada literal marcado $[L_h]$ da fórmula hospedeira. A cada ponto de backtracking está associada uma subfórmula ν_i, inserida na fórmula hospedeira como resultado de um passo de inferência.

No exemplo 4.17 a subfórmula ($E \wedge A$) está associada ao ponto de backtracking referente ao literal marcado [C].

***Exemplo 4.17*:**

$$A \rightarrow B \wedge (C \rightarrow D), \quad E \wedge A \rightarrow C \quad \vdash \quad A \rightarrow B \wedge ([C] \wedge (E \wedge A) \rightarrow D)$$

◆

Com o decorrer da prova, a meta contém vários pontos de backtracking com suas respectivas subfórmulas associadas.

Se uma fórmula visitante v_i falhar, devemos retornar à $[L_h]$ e substituí-la por uma outra fórmula visitante v_{i+1} .

Esgotadas todas as possíveis fórmulas visitantes, dizemos que ocorre uma falha no caminho da prova. Por conseguinte, respondendo à segunda questão: como conduzir a prova após a detecção da falha, devemos simplificar a fórmula hospedeira, desconsiderando $[L_h]$.

Na resolução proposicional isso é feito eliminando-se toda a cláusula onde ocorre o literal que falha. No MSEL não é possível eliminar toda a fórmula, uma vez que cada fórmula pode corresponder a várias cláusulas.

Seria correto simplesmente selecionar outro literal $[L_h']$ dentre os literais da subfórmula associada com o último ponto de backtracking. Todavia, para que se obtenha uma poda análoga à da resolução, procede-se a uma simplificação da fórmula hospedeira, visando eliminar o caminho de prova onde ocorre a falha.

A poda no ramo infrutífero é obtida substituindo-se o literal que falha pela constante lógica T e simplificando a fórmula resultante.

Exemplo 4.18:

1)	~E
2)	$A \rightarrow (C \wedge B)$
3)	$A \rightarrow E$
4) Goal	$D \rightarrow A$
5) de 4 e 2	$D \rightarrow [A] \wedge (C \wedge B)$
	Ocorre uma falha em b – substitui-se b por T
6)	$D \rightarrow [A] \wedge (C \wedge T)$
7)	$D \rightarrow [A] \wedge (C)$
	Ocorre uma falha em c – substitui-se c por T
8) de 7 e 1	$D \rightarrow [A] \wedge T$
	Todos os ramos de prova da fórmula visitante $A \rightarrow (C \wedge B)$ falharam
	Backtracking
9) de 4 e 3	$D \rightarrow [A] \wedge E$
10) 9 e 1	$D \rightarrow [a] \wedge \perp$
	$D \rightarrow \perp$
	~D
	Ocorre uma falha

◆

A simplificação exemplificada é somente um artifício para acelerar a prova. A rigor, para fins de completude, como desejamos mostrar que sempre existe uma prova, é suficiente selecionar um a um todos os literais da subfórmula associada com o ponto de backtracking .

4.6 - MSEL e Primeira Ordem

Esta seção apresenta o MSEL aplicado a fórmulas da lógica de primeira ordem livres de quantificadores.

Supõe-se que as fórmulas foram pré-processadas de forma que todas as variáveis quantificadas existencialmente foram devidamente traduzidas para funções de Skolem. Deixaram de ser escritos os quantificadores universais que regem todas as variáveis remanescentes.

As modificações necessárias no método para permitir fórmulas da lógica de primeira ordem, dizem respeito apenas às instanciações das variáveis e ao estabelecimento de um nível máximo que delimita o tamanho da prova. As modificações são simples, uma vez que se supõem resolvidas no pré-processamento as questões envolvendo os quantificadores.

Redefinindo a regra de inferência para lógica de primeira ordem, temos:

Definição 4.5

Sejam h e v fórmulas que possuem, respectivamente, literais L_h e L_v e seja ϕ o mais geral unificador que torna $L_h\phi$ e $L_v\phi$ complementares. A resultante r de h e v é a fórmula $h\,(\,L_h\phi /\!/ \,v\,(\,L_v\phi /\!/ \perp)\,)\,\phi$.

❐

O MSEL com eliminação de modelos também é redefinido facilmente, pela inclusão do mais geral unificador entre os literais.

Definição 4.6

Sejam *h* e *v* fórmulas que possuem, respectivamente, literais L_h e L_v e seja ϕ o mais geral unificador que torna $L_h\phi$ e $L_v\phi$complementares. A regra de inferência do MSEL com eliminação de modelos é dada por:

$$h, v \vdash \; r = h(L_h \phi // ([L_h\phi] \wedge v(L_v\phi // \perp)))\phi$$

□

Definição 4.7

Uma MSEL-dedução de uma formula β a partir de uma base de fórmulas $\alpha_1,\alpha_2,...\alpha_n$.é uma sequência de fórmulas $\beta_1,\beta_2, ...,\beta_m$, onde:

$\beta_m = \beta$,

β_1 é obtido por extensão de h e v através da regra acima tais que $h, v \in \{\alpha_1,\alpha_2,...\alpha_n\}$ e,

cada β_i é obtido por extensão de h e v tais que $h, v \in \{\alpha_1,\alpha_2,...\alpha_n, \beta_1,\beta_2, ...,\beta_{i-1}\}$

□

Definição 4.8

Uma MSEL- refutação é uma MSEL-dedução da constante $\perp$.

□

Com relação ao backtracking, a preocupação adicional com as fórmulas de primeira ordem é desfazer apropriadamente as unificações depois de cada falha. Evidentemente, devemos estabelecer também um limite para o tamanho da prova, que, se extrapolado, provoca o acionamento do backtracking.

O exemplo 4.19 mostra a refutação da meta ~q(a,b). No exemplo, estão sublinhadas as subfórmulas associadas com cada ponto de backtracking.

Exemplo 4.19:

1. $\sim q(a,b)$
2. $q(a,X) \vee r(b,X)$
3. $s(b) \wedge r(b,b) \rightarrow p(a)$
4. $s(X) \wedge r(b,X) \rightarrow p(X)$
5. $s(a)$
6. $s(b)$
7. $\sim p(b)$

8.	$\underline{[\sim q(a,b)] \wedge (\ r(b,b)\ )}$	de 1) e 2)
9.	$\underline{[\sim q(a,b)] \wedge (\underline{[r(b,b)] \wedge (s(b) \rightarrow p(a))})}$	de 8) e 3)
10.	$\underline{[\sim q(a,b)] \wedge (\underline{[r(b,b)] \wedge (\underline{[s(b)] \wedge p(a)})})}$	de 9) e 6)
11.	$\underline{[\sim q(a,b)] \wedge (\underline{[r(b,b)] \wedge (\underline{[s(b)] \wedge V})})}$	falha p(a)
12.	$\underline{[\sim q(a,b)] \wedge (\underline{[r(b,b)] \wedge (\perp \rightarrow p(a))})}$	falha s(b)
13.	$\underline{[\sim q(a,b)] \wedge (\underline{[r(b,b)] \wedge V})}$	falha (s(b) $\rightarrow$ p(a))
14.	$\underline{[\sim q(a,b)] \wedge (\underline{[r(b,b)] \wedge (s(b) \rightarrow p(b))})}$	após backtracking de 8) e 4)
15.	$\underline{[\sim q(a,b)] \wedge (\underline{[r(b,b)] \wedge (\underline{[s(b)] \wedge p(b)})})}$	de 14) e 6)
16.	$\underline{[\sim q(a,b)] \wedge (\underline{[r(b,b)] \wedge (\underline{[s(b)] \wedge \underline{[p(b)] \wedge \perp}})})}$	de 15) e 7)
17.	$\underline{[\sim q(a,b)] \wedge (\underline{[r(b,b)] \wedge (\underline{[s(b)] \wedge \perp})})}$	Redução
18.	$\underline{[\sim q(a,b)] \wedge (\underline{[r(b,b)] \wedge \perp})}$	Redução
19.	$\underline{[\sim q(a,b)] \wedge \perp)}$	Redução
20.	$\perp$	Redução

◆

5. SIMPLIFICAÇÃO POR ELIMINAÇÃO DE LITERAIS E OUTROS MÉTODOS DE PROVA

O presente capítulo aborda o relacionamento do MSEL com outros métodos de prova. Em particular, demonstra-se que o MSEL quando restrito a cláusulas, produz como resultante o mesmo resolvente obtido através da resolução. Em seguida, são abordados dois métodos que guardam algum relacionamento com o MSEL: o GCL/GCPL, de Avron (Avron,1993) e a NC-Resolution, de Murray (Murray,1982), (Murray,1988).

5.1 - Resolução

Nesta seção mostraremos a Resolução como um caso particular do Método de Simplificação por Eliminação de Literais. Para isso, vamos analisar o funcionamento do MSEL quando aplicado a cláusulas, e verificar que, nesse caso particular, o resolvente obtido via Resolução é exatamente o mesmo obtido utilizando-se o MSEL.

Como sabemos, para que duas cláusulas C_1 e C_2 sejam resolvidas, devem existir um literal L_1 em C_1, um literal L_2 em C_2, e um mais geral unificador ϕ tal que $L_1\phi$ e $L_2\phi$ são complementares. Ou seja $L_1\phi = \sim L_2\phi$.

Obtém-se a cláusula resolvente, C_R, eliminando-se $L_1\phi$ de $C_1\phi$, eliminando-se $L_2\phi$ de $C_2\phi$, e, finalmente, fazendo a disjunção das duas cláusulas remanescentes.

Sem perda de generalidade, sejam $C_1 = C_{R1} \vee L$ e $C_2 = C_{R2} \vee \sim L$, onde C_{R1} e C_{R2} são cláusulas (possivelmente vazias). Então temos como resolvente:

$$C_R = (C_{R1} \vee C_{R2})\phi.$$

Examinemos agora a fórmula resultante obtida utilizando-se o MSEL. Primeiramente, verificaremos, com a ajuda do lema 3.3, que, se uma fórmula γ está na forma clausal, o sinal de cada um de seus literais é igual à polaridade. Consequentemente, não há necessidade de ajustes de sinal correspondentes às fases 1.1 e 2.1 do MSEL.

$$C_1 , C_2 \vdash C_1 (L_h\phi // (C_2 (L_v\phi // \perp))))\phi = C_1 (L_h\phi / C_2 (L_v\phi / \perp)))\phi$$

Em seguida, ver-se-á, que o resultado das fases 1.2 e 2.1 corresponde ao resolvente obtido através da Resolução.

Na fase 1.2., seja $v = C_2 = C_{R2} \vee \sim L$ a fórmula visitante genérica na forma clausal, e seja $\sim L$ o literal L_2 a ser substituído pela constante lógica 'c' obtida na fase 1.1.

Já sabemos que $c = \perp$(falso). Portanto,

$v (L_2 / c) = v (\sim L / \perp) = R_2 \vee \perp$ Simplificando, obtemos:

$v_f = C_{R2}$

Temos que a fase 1.2 aplicada a cláusulas corresponde à eliminação do literal $\sim L$ da cláusula C_2, resultando $v_f = C_{R2}$.

Finalmente, na fase 2.2 temos a substituição de L_1 por v_{fa} em h.

Seja $h = C_1 = C_{R1} \vee L$ a fórmula hospedeira genérica na forma clausal, e seja L o literal L_1 a ser substituído pela simplificação da fórmula visitante com o sinal ajustado, v_{fa}, obtida na fase 2.1.

Sabemos que o resultado da fase 1 é $v_f = C_{R2}$, e que, de acordo com a fase 2.1, $v_{fa} = v_f = R_2$. Logo,

$$h\,(L_1 / v_{fa}) = h\,(L / C_{R2}) = C_{R1} \vee C_{R2}$$

Assim, o resultado do MSEL, $\psi = C_{R1} \vee C_{R2}$, corresponde ao mesmo passo obtido através da Resolução.

Exemplo 5.1

Sejam h e v as fórmulas indicadas abaixo:

h) $\overset{+}{A} \vee \sim \overset{-}{B} \vee \overset{+}{C}$ $\qquad L_1 = \overset{+}{C}$

v) $\sim \overset{-}{C} \vee \sim \overset{-}{D} \vee \overset{+}{E}$ $\qquad L_2 = \sim \overset{-}{C}$

---

r) $\overset{+}{A} \vee \sim \overset{-}{B} \vee \sim \overset{-}{D} \vee \overset{+}{E}$

Mostrando, passo a passo, a dedução, temos:

fase 1.1) cálculo de ***c***

$c = \perp$ (falso) $\qquad$ pois o sinal real de $L_2 = \sim \overset{-}{C}$ é igual a seu sinal aparente

fase 1.2) substituição de C por L_2, gerando v_f

$$v_f = \sim\overset{-}{C} \vee \sim\overset{-}{D} \vee \overset{+}{E}\ (\sim\overset{-}{C}/\bot) \quad = \quad F \vee \sim\overset{-}{D} \vee \overset{+}{E}$$

$$= \quad \sim\overset{-}{D} \vee \overset{+}{E}$$ nenhuma simplificação é possível

fase 2.1) ajuste de v_f

$v_{fa} = v_f = \sim\overset{-}{D} \vee \overset{+}{E}$ pois o sinal real de $L_1 = \overset{+}{C}$ é igual a seu sinal real

fase 2.2) substituição de L_1 em h e simplificação de $h\ (L_1 / v_{fa})$

$$h\ (L_1 / v_{fa}) = h\ (\overset{+}{C} / \sim\overset{-}{D} \vee \overset{+}{E}) = \overset{+}{A} \vee \sim\overset{-}{B} \vee \sim\overset{-}{D} \vee \overset{+}{E}$$

$\psi = \overset{+}{A} \vee \sim\overset{-}{B} \vee \sim\overset{-}{D} \vee \overset{+}{E}$ nenhuma simplificação é possível

Obviamente, $\psi = \overset{+}{A} \vee \sim\overset{-}{B} \vee \sim\overset{-}{D} \vee \overset{+}{E}$, é o mesmo resultado obtido via Resolução.

5.2 - Métodos Híbridos e Variações

Esta seção aborda dois métodos de prova que guardam algum relacionamento com o MSEL. O cálculo GCL/GCPL, de Avron, também procura utilizar uma combinação do método de Tableaux e Resolução, enquanto a NC-Resolution, de Murray, aplica a resolução a fórmulas não clausais.

5.2.1 - GCL e GCPL

Em (Avron,1993) resolução e Tableau são formalizados a partir de cálculos de seqüentes, como o proposto por Gentzen. São apresentadas duas versões do cálculo de seqüentes: uma versão proposicional GCPL (Gentzen Classical Propositional Logic) e uma extensão para lógica de primeira ordem, denominada GCP (Gentzen Classical Logic).

Avron argumenta que encontrar a cláusula vazia [2], a partir de um conjunto de seqüentes, é um problema difícil, tanto utilizando o método de Tableaux quanto utilizando resolução. No contexto proposicional (GCPL), considera o Tableau preferível à resolução, enquanto, para lógica clássica, afirma que, em função do número de ramificações para cada instanciação possível, "O método de Tableau 'clássico' não é prático para nada que não seja trivial".

---

[2] Uma cláusula é definida como "um seqüente que consiste apenas de fórmulas atômicas".

Assim, sugere combinar os dois métodos, utilizando primeiramente o método de Tableaux, e, em seguida, a resolução. O Tableau seria utilizado para:

a) obter cláusulas de maneira eficiente

b) dividir em tarefas menores o trabalho a ser feito pela resolução – cada ramo do Tableau pode ser considerado um sub-problema .

O exemplo 5.2 mostra a utilização do Tableau para a geração de um conjunto de cláusulas equivalentes a $((A \vee B) \wedge (A \vee C) \rightarrow A \vee (B \wedge C))$.

Exemplo 5.2

$$T\,((A \vee B) \wedge (A \vee C) \rightarrow A \vee (B \wedge C))$$

$F\,((A \vee B) \wedge (A \vee C))$		$T\,(A \vee (B \wedge C))$	
$F\,(A \vee B)$	$F\,(A \vee C)$	TA	TB
FA	FA		TC
FB	FC		

◆

Tomando os literais de cada ramo do Tableau, são geradas as cláusulas { { ⇒A, B }, {⇒A, C }, { A⇒ }, { B,C⇒ } }. A partir desse conjunto de cláusulas, a cláusula vazia {⇒} pode ser gerada utilizando-se resolução.

Avron coloca como uma questão a ser pesquisada os critérios e heurísticas para indicar até onde ir utilizando Tableau e de onde prosseguir utilizando a resolução.

O GCL consiste numa utilização conjunta do método de Tableaux seguido de aplicação da resolução. O MSEL, diferentemente, é um método que usa fórmulas não clausais como o método de Tableaux mas com as heurísticas da Resolução. O MSEL não usa independentemente nem o método do tableaux nem a resolução como faz o GCL.

5.2.2 - NC-Resolution

Em (Murray,1982) é apresentado um procedimento de prova para fórmulas livres de quantificadores do cálculo de predicados.

O procedimento de prova denominado NC-Resolution (Non Clausal Resolution) parte de duas fórmulas bem formadas quaisquer α e β, e deduz um NC-resolvente de α e β.

A fórmula α deve possuir um literal L_α, e a fórmula β , outro átomo L_β tais que L_α e L_β ocorrem com polaridades opostas e podem ser unificados por um unificador mais geral ϕ , isto é, $L_\alpha\phi = L_\beta\phi = L$.

Supondo que L_α é positivo em α ,e que L_β é negativo em β, o NC-resolvente de α e β é definido como:

$$\alpha\,\phi\,\{\perp / L\} \vee \beta\,\phi\,\{\top / L\}$$

Caso contrário, se L_α é negativo em α e L_β é positivo em β, o NC-resolvente de α e β é definido como:

$$\alpha\,\phi\,\{\top / L\} \vee \beta\,\phi\,\{\perp / L\}$$

Exemplo 5.3

Seja $\alpha = \sim p(X) \rightarrow q(X)$, onde o átomo p(X) ocorre positivamente e seja $\beta = p(a) \rightarrow r(b)$, onde o átomo p(a) ocorre negativamente. Nesse caso o unificador mais geral ϕ corresponde a $\{X/a\}$.

Temos, então, que o NC–resolvente de α e β é:

$$(\sim p(a) \rightarrow q(a)\ \{\perp / p(a)\}) \vee (\ p(a) \rightarrow r(b)\ \{\top / (a)\}\ )$$

$$(\sim\perp\rightarrow q(a)\,)\ \vee\ (\ \top\rightarrow r(b)\ )$$

$$(a)\vee r(b)$$ ◆

Segundo Murray, a NC-Resolution é uma simplificação da regra de inferência

$$(\alpha\phi\{\top/L\}\wedge\beta\phi\{\top/L\})\quad\vee\quad(\alpha\phi\{\perp/L\}\wedge\beta\phi\{\perp/L\})$$

apresentada em notas de aula por E.F. STORM na Universidade de Siracusa, em 1972.

O método de Storm corresponde à idéia de que L será verdadeiro ou falso. Se L é verdadeiro, podemos substituir sua ocorrência por $\top$, isto é, $(\alpha\phi\{\top/L\}\wedge\beta_j\phi\{\top/L\})$. Alternativamente, se L for falso, substituímos as ocorrências de L por $\perp$, ou seja, $(\alpha\phi\{\perp/L\}\wedge\beta\phi\{\perp/L\})$. Considerando ambas as alternativas, temos a regra de inferência acima.

Para simplificar a regra de inferência de Storm, vamos primeiramente convertê-la para forma normal conjuntiva disjuntiva. Obtemos, então:

$\alpha\phi\{\perp/L\}\vee\beta\phi\{\top/L\}$ [1]

$\alpha\phi\{\top/L\}\vee\beta\phi\{\perp/L\}$ [2]

$\alpha\phi\{\top/L\}\vee\beta\phi\{\perp/L\}$ [3]

$\alpha\phi\{\top/L\}\vee\beta\phi\{\perp/L\}$ [4]

Qualquer das quatro fórmulas acima, isoladamente ou em conjunto, pode ser deduzida a partir de α e β.

Observamos que na fórmula [3] temos ocorrências apenas de α. A fórmula [3] é expletiva, uma vez que é subsumida por α. De forma análoga, a fórmula [4] contém ocorrências apenas de β. Consequentemente, também deve ser desprezada.

A NC-Resolution leva em conta as polaridades opostas de L em α e β e seleciona entre as fórmulas [1] e [2].

A NC-Resolution é o método que mais se assemelha ao MSEL, principalmente no que se refere a utilização da polaridade como critério de seleção dos literais que serão resolvidos. Todavia, suas regras de inferência sempre produzem disjunções como resultante, enquanto o MSEL preserva os conectivos utilizados pelo usuário. Na NC-Resolution, a questão da falha também não é abordada, nem são cogitadas novas heurísticas.

6. CORRETUDE e COMPLETUDE do MSEL

Para provar a corretude do MSEL, vamos desenvolver duas provas. A primeira, mais intuitiva, faz referência ao método de Tableaux. Provaremos que a inferência segundo o MSEL corresponde a aplicação de três operações sobre o método de Tableaux. Como cada uma das operações é correta, a aplicação das três também o será.

A segunda prova da corretude, por indução, visa dar cores próprias ao MSEL, salientando, num sentido de independência, que este não precisa se apoiar no método de Tableaux.

Para prova da completude, mostramos que, se conseguimos deduzir a constante lógica $\perp$ através da Resolução, então, também o conseguiremos utilizando o MSEL. Consequentemente, se a Resolução é completa, o MSEL também o será.

Para a primeira prova da corretude, primeiramente vamos examinar algumas propriedades de Tableaux não fechados que nos serão úteis.

6.1. Propriedades de Tableaux Abertos

Notemos que para uma dada fórmula vários Tableaux são possíveis. Por exemplo, para a fórmula $(A \vee B) \wedge (C \vee D)$, os dois Tableaux abaixo são possíveis, conforme se escolha expandir inicialmente a primeira ou a segunda das disjunções.

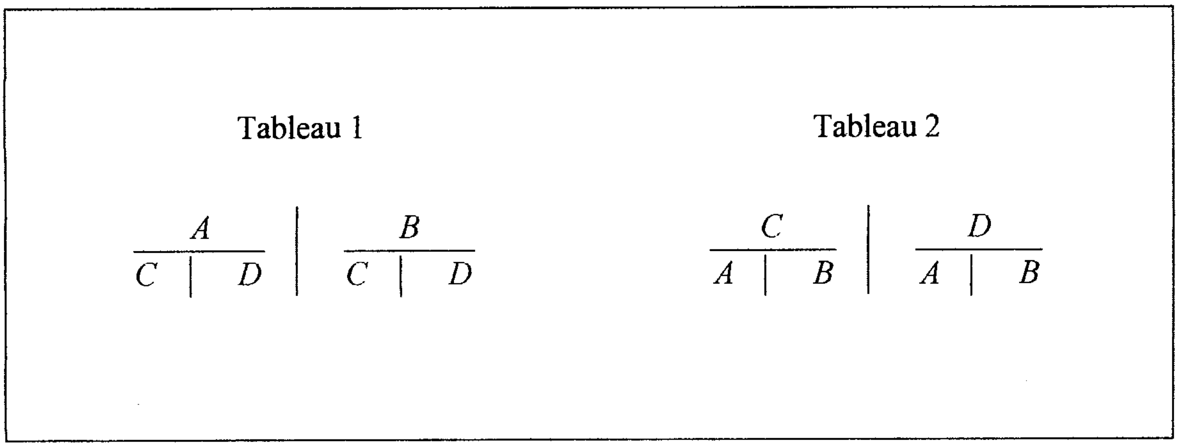


Figura 6.1 - Exemplo de dois Tableaux para a fórmula $(a \vee b) \wedge (c \vee d)$

No Tableau 1 da Figura 6.1, dizemos que 'A' e 'B' estão em nós do Tableau, enquanto os literais 'C' e 'D' aparecem nas folhas. No Tableau 2, inversamente, 'C' e 'D' aparecem em nós, enquanto 'A' e 'B' ocorrem nas folhas.

Para uma fórmula h e para um literal qualquer L pertencente a h, sempre é possível escrever uma árvore Tableau de forma que L ocorra numa folha da árvore.

Agora definiremos, para uma fórmula h e para um literal L de h, um Tableau (TableauE(L) de h) em que L ocorre numa folha.

Definição 6.1

Chamaremos de Tableau de h construído em função de L (TableauE(L) de h), o Tableau da fórmula h construído expandindo somente as subfórmulas de h onde L ocorre.

❐

Exemplo 6.1

Assim, para h=((A→B)∨D)∧(C∨B) e para L= A, teríamos as seguintes expansões:

1) Fórmula Inicial

$$((A \rightarrow B) \vee D) \wedge (C \vee B)$$

2) Aplicando a regra tipo α na subfórmula de $((A \rightarrow B) \vee D) \wedge (C \vee B)$ que contém o literal L=A, temos:

$$\begin{array}{c} (A \rightarrow B) \vee D \\ (C \vee B) \end{array}$$

3) Aplicando a regra tipo β na subfórmula de (A→B)∨D que contém o literal L=A, temos:

$$\begin{array}{c|c} \multicolumn{2}{c}{(C \vee B)} \\ \hline (A \rightarrow B) & D \end{array}$$

4) Aplicando a regra tipo β na subfórmula (A→B) que contém o literal L=A, temos:

$$\begin{array}{c|c|c} \multicolumn{3}{c}{(C \vee B)} \\ \hline \sim A & B & D \end{array}$$

que é o TableauE(A) de =((A→B)∨D)∧(C∨B).

◆

Note que no TableauE(L) de h:

a) não foram aplicadas as expansões referentes às subfórmulas onde L não ocorre;

b) o literal L sempre está numa folha.;

c) o literal L ocorre na folha com o sinal aparente igual ao sinal real.

Precisamos verificar outra propriedade dos Tableaux referente à eliminação de literais.

Lema 6.1

Seja uma fórmula h do tipo $\alpha_1 \wedge(\alpha_2 \vee(L\wedge\alpha_3))$,onde α_3 possui pelo menos um literal então h tem Tableau da forma

$$\begin{array}{c|c} \multicolumn{2}{c}{T'} \\ \hline T'' & \begin{array}{c} L \\ T''' \end{array} \end{array}$$

onde: T', T'', T''' são Tableaux respectivamente para α_1, α_2 e α_3

Então, para qualquer fórmula β, do tipo $\alpha_1 \wedge (\alpha_2 \vee \alpha_3)$ cujo tableau é obtido do tableau de h pela eliminação do literal L

$$\begin{array}{c|c} \multicolumn{2}{c}{T'} \\ \hline T'' & T''' \end{array}$$

, temos que se $\models h$ então $\models \beta$.

Prova:

A justificativa desse Lema é simples. Basta notar que se $\models \alpha_1 \wedge(\alpha_2 \vee(L\wedge\alpha_3))$ então $\models \alpha_1 \wedge(\alpha_2 \vee \alpha_3)$.

❑

Exemplo 6.2:

Tomemos a fórmula $A\rightarrow(B\wedge C)$, cujo Tableau é $\begin{array}{c|c} \sim A & \begin{array}{c} B \\ C \end{array} \end{array}$

Eliminando o literal 'C' do Tableau acima, obtemos: $\begin{array}{c|c} \sim A & B \end{array}$

que é Tableau para $A \rightarrow B$. Claramente, $A \rightarrow (B \wedge C) \vdash A \rightarrow B$.

Eliminando o literal 'B' do Tableau acima, obtemos: $\sim A \mid C$

que é para $A \rightarrow C$. Claramente, $A \rightarrow (B \wedge C) \vdash A \rightarrow C$.

◆

6.2 Correspondência entre as fases do MSEL e o Método de Tableaux

O método de Eliminação de Literais leva em conta a existência de várias fórmulas numa base de conhecimento. Em cada prova, apenas as fórmulas apropriadas serão selecionadas e incorporarão a árvore de prova.

Vamos agora examinar como, a partir de uma fórmula meta h e de uma fórmula β pertencente à base de conhecimento, conseguimos a resultante r, tal que h , $\beta \vdash r$

Sejam h e β duas fórmulas que possuem , respectivamente, literais L e ~L de polaridade complementares e cujo mais geral unificador é ϕ. Vamos selecionar arbitrariamente a fórmula h como hospedeira e β como visitante.

Seja T_hosp o TableauE(L) de h e T_visit o TableauE(~L) de β. Vamos determinar, a partir de T_hosp e T_visit, dois Tableaux intermediários T_r1 e T_r2. Mostraremos que T_r2 é Tableau da fórmula $h \wedge \beta$. E, a partir de T_r2, vamos determinar T_result, que é Tableau de uma fórmula r, tal que $h \wedge \beta \vdash r$.

Como, de h, $\beta \vdash h \wedge \beta$, a corretude do MSEL é justificada pela corretude do método de Tableaux.

Para obter T_result, vamos aplicar as seguintes operações:

1) obter T_r1 enxertando T_visit no ramo de T_hosp onde L ocorre;

2) obter T_r2 fechando o ramo de T_r1 onde ocorrem L e ~L. No caso de fórmulas de lógica de primeira ordem livres de quantificadores, a única preocupação adicional é a de instanciar as variáveis de acordo com o mais geral unificador de L e ~L;

3) obter T_result eliminando a ocorrência de L em T_r2.

Exemplo 6.3:

Seja $h = (\overset{-}{A} \vee \sim (\overset{-}{B} \to \overset{+}{C})) \to \overset{+}{D}$ e seja L= $\overset{-}{B}$. Então o TableauE($\overset{-}{B}$) de h

T_hosp é: $\begin{array}{c|c} \dfrac{\sim A}{\sim B \mid C} & D \end{array}$

Seja $\beta = (\overset{-}{G} \to \overset{+}{B}) \wedge \overset{+}{E} \wedge \overset{+}{D}$ e seja ~L= $\overset{+}{B}$. Então o TableauE($\overset{+}{B}$) de β

T_visit , é: $\dfrac{E \wedge D}{\sim G \mid D}$

1) enxertando T_visit no ramo de T_hosp onde $\bar{B}$ ocorre, teremos:

$$T\_r1 = \begin{array}{c|c} \begin{array}{c} \sim A \\ \hline \begin{array}{c|c} \begin{array}{c} \sim B \\ \\ E \wedge D \\ \begin{array}{c|c} \sim G & B \end{array} \end{array} & C \end{array} \end{array} & D \end{array}$$

Note que T_r1 é Tableau para a fórmula $h \wedge \beta$, onde a fórmula β foi expandida num ramo em que h tinha sido expandida previamente.

2) Agora vamos fechar o ramo onde B e ~B ocorrem. Naturalmente, esta é também uma operação correta no método de Tableaux.

$$T\_r2 = \begin{array}{c|c} \begin{array}{c} \sim A \\ \hline \begin{array}{c|c} \begin{array}{c} \sim B \\ E \wedge D \\ \sim G \end{array} & C \end{array} \end{array} & D \end{array}$$

Veja que a fórmula visitante foi enxertada num ramo da fórmula hospedeira, onde considera-se ~B verdadeiro. Nesse contexto, o literal B da fórmula visitante é falso. Daí a substituição de B por $\perp$.

3) A operação seguinte é eliminar o literal L de T_r2 . Então, teremos:

$$T\_result = \begin{array}{c|c} \begin{array}{c} \sim A \\ \hline \begin{array}{c|c} \begin{array}{c} E \wedge D \\ G \end{array} & C \end{array} \end{array} & D \end{array}$$

A operação acima corresponde à substituição de ~B, em T_hosp, pela simplificação de T_visit.

◆

No Método Simplificação por Eliminação de Literais observamos uma inversão na ordem em que são aplicadas as regras 1) e 2) . A operação 2), é realizada antes da operação 1) .

A operação 2) - fechamento de T_r1 -corresponde a fase 1 do MSEL, qual seja a substituição do literal L por uma constante lógica e a simplificação, eliminando o ramo inconsistente.

A operação 1) ,enxerto de T_visit dentro de T_hosp, juntamente com a operação 3) , eliminação da ocorrência de L em T_r2., corresponde à fase 2 do MSEL.

Analisando as operações 1) e 2) vemos que:

A operação 1) é correta pois trata-se da expansão de β num ramo do Tableau iniciado por h.

A operação 2) também é correta pois trata-se do fechamento de um dos ramos do Tableau obtido pela operação 1).

Vamos mostrar que T_result , obtido pela operação 3), é Tableau de uma fórmula r, tal que $h \wedge \beta \vdash r$.

T_result é obtido de T_r2 , Tableau de $h \wedge \beta$ pela eliminação de L.

T_r2 é o resultado do enxerto da simplificação de T_visit dentro da folha de T_hosp que contém L. Temos que T_r2 é da forma:

$$\begin{array}{c|c} \multicolumn{2}{c}{T\_hosp'} \\ \hline & T\_hosp''' \\ T\_hosp'' & L \\ & T\_visit \end{array}$$

Fazendo, pelo Lema 6.2, a eliminação de L, temos:

$$\text{T\_result} = \begin{array}{c|c} \multicolumn{2}{c}{T\_hosp'} \\ \hline T\_hosp'' & \begin{array}{c} T\_hosp''' \\ T\_visit \end{array} \end{array}$$

Logo, como T_r2 é Tableau para $h \wedge \beta$, então, para qualquer fórmula r cujo Tableau é T_result, temos que $h \wedge \beta \vdash r$.

6.3 - Corretude do MSEL

Esta seção visa provar a corretude do MSEL, utilizando indução no número de níveis de cada fórmula.

Para facilitar a prova de corretude do método, vamos usar o artifício de transformar as fórmulas para Negation Normal Form.

Primeiramente, provaremos a equivalência da transformação aplicada, e, em seguida, provaremos a corretude do método para as fórmulas traduzidas para a forma normal.

6.3.1 - MSEL aplicado a Fórmulas em Negation Normal Form.

A finalidade desta seção é provar que o resultado da aplicação do MSEL a duas fórmulas h e v quaisquer produz, como resultado, fórmula equivalente àquela obtida pela aplicação do MSEL à Negation Normal Form de h e v.

Seja $\mathcal{N}$ a função que dá a Negation Normal Form de uma fórmula, e, durante essa prova, seja $\vdash$ a dedução utilizando o MSEL. Temos então o seguinte teorema.

***Teorema 6.1*:**

$h,v \vdash r$ se e somente se $\mathcal{N}(h), \mathcal{N}(v) \vdash \mathcal{N}(r)$.

Prova

Como sabemos, toda fórmula tem uma equivalente em Negation Normal Form ($h \leftrightarrow \mathcal{N}(h)$),

logo : $h \leftrightarrow \mathcal{N}(\mathrm{h})$,

$v \leftrightarrow \mathcal{N}(v)$ e

$r \leftrightarrow \mathcal{N}(r)$.

Assim sendo , se $\mathcal{N}(h), \mathcal{N}(v) \vdash \mathcal{R}$,basta provar que $\mathcal{R} = \mathcal{N}(r)$.

Ora, para dois literais complementares: L_h de $\mathcal{N}(h)$ e L_v de $\mathcal{N}(v)$, onde ϕ é o unificador mais geral de L_h e L_v ,

$$\mathcal{R} = \mathcal{N}(h) (L_h \; // \; \mathcal{N}(v) (L_v // \perp))) \phi$$

Mas , pelo Lema 3, como $\mathcal{N}(h)$ e $\mathcal{N}(v)$ estão na negation normal form ,temos:

$$\mathcal{R} = \mathcal{N}(h) (L_h \; / \; \mathcal{N}(v) (L_v / \perp))) \phi$$

Por outro lado,

$$\mathcal{N}(r) = \mathcal{N}(h (L_h \; // \; v (L_v // \perp)) \phi$$

e, pelo Lema 6: $\mathcal{N}(h (L_h // \beta)) = \mathcal{N}(h) (L_\alpha / \mathcal{N}(\beta))$. Então:

$$\mathcal{N}(r) \;\; = \mathcal{N}(h) (L_h \; / \; \mathcal{N}(v (L_v // \perp))) \phi =$$

$$= \mathcal{N}(h) (L_h \; / \; \mathcal{N}(v) (L_v / \mathcal{N}(\perp)))) \phi =$$

$$= \mathcal{N}(h) (L_h \; / \; \mathcal{N}(v) (L_v / \perp))) \phi = \mathcal{R}$$

❑

6.3.2 - Corretude para Fórmulas Em NegationNormal Form

Sejam h e v fórmulas em Negation Normal Form, e L um literal de h cujo literal complementar $\sim L$ ocorre em v.

Sabemos, por definição, que: $h , v \vdash h(L / v(\sim L / \perp)) \phi$. Desejamos provar que $h(L / v(\sim L / \perp)) \phi$ é conseqüência lógica de h e v.:

Teorema 6.2

$$\models h \wedge v \rightarrow h(L / v(\sim L / \perp))\,\phi$$

Prova

Seja m o nível de L em h, e

seja n o nível de $\sim L$ em v.

Então , h e v são da forma:

$$h = h_m = \alpha_m \wedge (\beta_m \vee (\alpha_{m-1} \wedge (\beta_{m-1} \vee \ldots (\alpha_1 \wedge (\beta_1 \vee (\alpha_0 \wedge L)) \ldots))),$$

onde:

α_x é da forma $\beta_{x1} \wedge \beta_{x2} \wedge \beta_{x3} \wedge \ldots \wedge \beta_{xz}$,

β_x é da forma $\alpha_{x1} \vee \alpha_{x2} \vee \alpha_{x3} \vee \ldots \vee \alpha_{xz}$ e

α_m e α_0 podem ser vazios.

Note que: $h_m = \alpha_m \wedge (\beta_m \vee h_{m-1})$

Obviamente, v tem a mesma forma de h. Todavia, para fazermos distinção entre as subfórmulas de h e v, vamos representar v da forma abaixo (substituindo α por γ e β por ω):

$$v = v_n = \gamma_n \wedge (\omega_n \vee (\gamma_{n-1} \wedge (\omega_{n-1} \vee \ldots (\gamma_1 \wedge (\omega_1 \vee (\gamma_0 \wedge \sim L)) \ldots))),$$

onde:

γ_x é da forma $\omega_{x1} \wedge \omega_{x2} \wedge \omega_{x3} \wedge \ldots \wedge \omega_{xz}$,

ω_x é da forma $\gamma_{x1} \vee \gamma_{x2} \vee \gamma_{x3} \vee \ldots \vee \gamma_{xz}$ e

γ_n e γ_0 podem ser vazios.

note que : $v_n = \gamma_n \wedge (\omega_n \vee v_{n-1})$

Faremos uma indução em função dos níveis m e n.

1) Para $m = 0$ e $n = 0$

$h_0 = \alpha_0 \wedge L$ e

$v_0 = \gamma_0 \wedge \sim L$.

Se existe um unificador mais geral ϕ de L e ~L, então, claramente, h_0 , $v_0 \vdash \perp$ e $h_0 \wedge v_0 \rightarrow \perp$

2) Para $m > 0$

Sejam:

$h_m = \alpha_m \wedge (\beta_m \vee (\alpha_{m-1} \wedge (\beta_{m-1} \vee \ldots (\alpha_1 \wedge(\beta_1 \vee (\alpha_0 \wedge L)) \ldots)))$ e

v_n.

se $h_{m-1} \wedge v_n \rightarrow r_{m-1,n}$

temos:

1) $h_m = \alpha_m \wedge (\beta_m \vee h_{m-1})$

2) v_n

3) $h_{m-1} \wedge v_n \rightarrow r_{m-1,n}$

---

4) α_m de 1)

5) $\beta_m \vee h_{m-1}$ de 1)

6) $\sim (h_{m-1} \wedge v_n) \vee r_{m-1,n}$ de 3)

7) $\sim h_{m-1} \vee \sim v_n \vee r_{m-1,n}$ de 6)

8) $\sim h_{m-1} \vee r_{m-1,n}$ de 7) e 2)

9) $\beta_m \vee r_{m-1,n}$ de 8) e 5)

10) $\alpha_m \wedge (\beta_m \vee r_{m-1,n})$ de 9) e 4)

então: $h_m \wedge v_n \rightarrow r_{m,n} = \alpha_m \wedge (\beta_m \vee r_{m-1,n})$

desenvolvendo:

se: $h_o \wedge v_n \rightarrow r_{0,n}$

então: $h_m \wedge v_n \rightarrow r_{m,n} = \alpha_m \wedge (\beta_m \vee (\alpha_{m-1} \wedge (\beta_{m-1} \vee \ldots (\alpha_1 \wedge (\beta_1 \vee (r_{0,n})) \ldots)))$

todavia, como $h_o = \alpha_0 \wedge L$, temos também:

$$h_o \wedge v_n \rightarrow \alpha_0 \wedge r_{0,n}$$

e finalmente:

$$h_m \wedge v_n \rightarrow \alpha_m \wedge (\beta_m \vee (\alpha_{m-1} \wedge (\beta_{m-1} \vee \ldots (\alpha_1 \wedge (\beta_1 \vee (\alpha_0 \wedge r_{0,n})) \ldots)))$$

a expressão acima é o resultado da substituição de L por $r_{0,n}$, ou seja:

i) $h_m \wedge v_n \rightarrow h_m(L / r_{0,n})$

3) Para $n > 0$ e $m = 0$.

Sejam:

$$v_n = \gamma_n \wedge (\omega_n \vee v_{n-1})$$

$$v_n = \gamma_n \wedge (\omega_n \vee (\gamma_{n-1} \wedge (\omega_{n-1} \vee \ldots (\gamma_1 \wedge (\omega_1 \vee (\gamma_0 \wedge \sim L)) \ldots))),$$

$$h_o$$

se :

$$h_0 \wedge v_{n-1} \rightarrow r_{0,n-1}$$

temos:

1) $v_n = \gamma_n \wedge (\omega_n \vee v_{n-1})$

2) h_0

3) $h_0 \wedge v_{n-1} \rightarrow r_{0,n-1}$

---

4)	γ_n	de 1)
5)	$\omega_n \vee v_{n-1}$	de 1)
6)	$\sim(h_0 \wedge v_{n-1}) \vee r_{0,n-1}$	de 3)
7)	$\sim h_0 \vee \sim v_{n-1} \vee r_{0,n-1}$	de 6)
8)	$\sim v_{n-1} \vee r_{0,n-1}$	de 7) e 2)
9)	$\omega_n \vee r_{0,n-1}$	de 8) e 5)
10)	$\gamma_n \wedge (\omega_n \vee r_{0,n-1})$	de 9) e 4)

então:

$$h_o \wedge v_n \rightarrow \gamma_n \wedge (\omega_n \vee r_{0,n-1})$$

desenvolvendo:

como: $h_0 \wedge v_0 \rightarrow \perp$

então: $h_o \wedge v_n \rightarrow r_{0,n} = \gamma_n \wedge (\omega_n \vee (\gamma_{n-1} \wedge (\omega_{n-1} \vee \ldots (\gamma_1 \wedge (\omega_1 \vee \perp) \ldots))),$

todavia, como $v_0 = \gamma_0 \wedge \sim L$, temos também:

$$h_0 \wedge v_0 \rightarrow \gamma_0 \wedge \perp$$

e finalmente:

$$h_o \wedge v_n \rightarrow \gamma_n \wedge (\omega_n \vee (\gamma_{n-1} \wedge (\omega_{n-1} \vee \ldots (\gamma_1 \wedge (\omega_1 \vee (\gamma_0 \wedge \perp) \ldots))).$$

A expressão acima é o resultado da substituição de $\sim L$ por $\perp$, ou seja:

ii) $h_o \wedge v_n \rightarrow v_n(\sim L / \perp)$

a partir dos resultados intermediários i) e ii) temos:

$h_m \wedge v_n \rightarrow h_m(L / v_n(\sim L / \perp))$

❑

6.4 - Completude Do MSEL

Vamos provar a completude do MSEL em função da completude da Resolução. A cada passo da prova, o resolvente pertence ao conjunto das cláusulas geradas a partir do resultante segundo o MSEL. Se uma contradição é gerada através da Resolução, também será gerada uma fórmula contraditória pelo MSEL.

Definição 6.2

Definimos CLAU (h) como o conjunto das cláusulas obtidas a partir de uma fórmula h.

❑

Definição 6.3

Se Γ é um conjunto de fórmulas $\alpha_1, \alpha_2, \ldots, \alpha_n$, definimos

$\mathrm{CLAU}(\Gamma) = \mathrm{CLAU}(\alpha_1) \cup \mathrm{CLAU}(\alpha_2) \cup \ldots \cup \mathrm{CLAU}(\alpha_n)$

❑

Exemplo 6.4

h	***CLAU(h)***
$A \rightarrow B$	$\{\sim A \vee B\}$
$A \rightarrow (B \wedge C)$	$\{\sim A \vee B, \sim A \vee C\}$
B	$\{B\}$

◆

Lema 6.2

Seja α uma fórmula da lógica de primeira ordem livre de quantificadores, então:

CLAU(α) = CLAU ($\mathcal{N}$ (α))

Prova:

O algoritmo de transformação para negation normal form pode ser visto como uma sub-rotina do procedimento para transformação para forma clausal.

Se α é livre de quantificadores, os passos do algoritmo de transformação para forma clausal referentes a skolemização e renomeação de variáveis terão sido realizados pelo usuário. A transformação para negation normal form terá eliminado as implicações e as negações que não ocorriam em nível de literal. O passo restante é a distribuição das conjunções e disjunções gerando as cláusulas.

❑

Lema 6.3

Se $C_1 \in$ CLAU(h) e $C_2 \in$ CLAU (v) e C_R é resolvente de C1 e C2, onde L_h é o literal resolvido em C_1e L_v o literal de C_2 (L_h e L_v são complementares e ϕ é o m.g.u de L_h e L_v), então:

$\exists$ r tal que $h, v \vdash r$ e $C_R \in$ CLAU(r)

Exemplo 6.1

h: $A \to (B \wedge F)$ CLAU (h) = { $\sim A \vee B$, $\sim A \vee F$ }

v: $D \to (A \vee E)$ CLAU (v) = { $\sim D \vee A \vee E$ }

Seja $C_1 = \sim A \vee F$, onde $L_h = \sim A$ e

$C_2 = \sim D \vee A \vee E$, onde $L_v = A$. Portanto, temos como resolvente:

$C_R = \sim D \vee E \vee F$

Ora, L_h ocorre em h com polaridade negativa e L_v ocorre com polaridade oposta. Então,

$\exists$ r tal que $h, v \vdash r$

em nosso exemplo, $r = \overbrace{D \to (\overset{+}{\underset{\perp}{A}} \vee E)}^{\overset{-}{A}} \to B \wedge F$

r: $\sim(D \to E) \to (B \wedge F)$ CLAU(r) = { $\sim D \vee E \vee B$, $\sim D \vee E \vee F$ }

Portanto, $C_R \in$ CLAU(r) ◆

Prova:

Sabemos que:

$$h, v \vdash r \quad \text{se e somente se} \quad \mathcal{N}(h), \mathcal{N}(v) \vdash \mathcal{N}(r).$$

Sejam L_h e L_v dois literais complementares de $\mathcal{N}(h)$ e $\mathcal{N}(v)$, cujo m.g.u é ϕ.

Seja m o nível de L_h em $\mathcal{N}(h)$, e

seja n o nível de L_v em $\mathcal{N}(v)$.

Então , $\mathcal{N}(h)$ e $\mathcal{N}(v)$ são da forma:

$\mathcal{N}(h)$ $\quad \mathcal{N}(h_m) = \alpha_m \wedge (\beta_m \vee (\alpha_{m-1} \wedge (\beta_{m-1} \vee \ldots (\alpha_1 \wedge(\beta_1 \vee (\alpha_0 \wedge L_h)) \ldots))),$

onde:

α_x é da forma $\beta_{x1} \wedge \beta_{x2} \wedge \beta_{x3} \wedge \ldots \wedge \beta_{xz}$,

β_x é da forma $\alpha_{x1} \vee \alpha_{x2} \vee \alpha_{x3} \vee \ldots \vee \alpha_{xz}$, e

α_m e α_0 podem ser vazios.

as cláusulas de $\mathcal{N}(h)$ que contêm L_h serão geradas a partir da distribuição dos literais de $\beta_m \vee \beta_{m-1} \vee \ldots \beta_1 \vee L_h$.

Obviamente, $\mathcal{N}(v)$ tem a mesma forma de $\mathcal{N}(h)$ vez que ambos estão na negation normal form. Todavia, para fazermos distinção entre as subfórmulas de $\mathcal{N}(h)$ e $\mathcal{N}(v)$, vamos representar $\mathcal{N}$(v) da forma abaixo (substituindo α por γ e β por ω) :

$\mathcal{N}(v)$ $\quad \mathcal{N}(v_n) = \gamma_n \wedge (\omega_n \vee (\gamma_{n-1} \wedge (\omega_{n-1} \vee \ldots (\gamma_1 \wedge (\omega_1 \vee (\gamma_0 \wedge L_v)) \ldots))),$

onde:

γ_x é da forma $\omega_{x1} \wedge \omega_{x2} \wedge \omega_{x3} \wedge \ldots \wedge \omega_{xz}$,

ω_x é da forma $\gamma_{x1} \vee \gamma_{x2} \vee \gamma_{x3} \vee \ldots \vee \gamma_{xz}$ e

γ_n e γ_0 podem ser vazios.

as cláusulas de $\mathcal{N}(v)$ que contêm L_v serão geradas a partir da distribuição dos literais de

$\omega_n \vee \omega_{n-1} \vee \ldots \omega_1 \vee L_v$,

Então os resolventes das cláusulas de $\mathcal{N}(v)$ e $\mathcal{N}(\mathrm{h})$ serão gerados a partir da distribuição dos literais de:

$$(\beta_m \vee \beta_{m-1} \vee \ldots \beta_1 \vee \omega_n \vee \omega_{n-1} \vee \ldots \omega_1)\,\phi$$

Ou seja, se $C_1 \in \mathrm{CLAU}(\mathcal{N}(h))$ e $C_2 \in \mathrm{CLAU}(\mathcal{N}(v))$, então:

$$C_R \in \mathrm{CLAU}\,((\beta_m \vee \beta_{m-1} \vee \ldots \beta_1 \vee \omega_n \vee \omega_{n-1} \vee \ldots \omega_1)\,\phi$$

Temos também que: $\mathcal{N}(r) = \mathcal{N}(h)\,(L_h \,/\, \mathcal{N}(v)\,(L_v / \perp)))\,\phi =$

$$\mathcal{N}(r) = \quad \alpha_m \wedge (\beta_m \vee (\alpha_{m-1} \wedge (\beta_{m-1} \vee \ldots (\alpha_1 \wedge (\beta_1 \vee (\alpha_0 \wedge$$

$$\gamma_n \wedge (\omega_n \vee (\gamma_{n-1} \wedge (\omega_{n-1} \vee \ldots (\gamma_1 \wedge (\omega_1 \vee (\gamma_0 \wedge \perp)) \ldots))),$$

$$)) \ldots)))\,\phi =$$

$$= \quad \alpha_m \wedge (\beta_m \vee (\alpha_{m-1} \wedge (\beta_{m-1} \vee \ldots (\alpha_1 \wedge (\beta_1 \vee (\alpha_0 \wedge$$

$$\gamma_n \wedge (\omega_n \vee (\gamma_{n-1} \wedge (\omega_{n-1} \vee \ldots (\gamma_1 \wedge (\omega_1)) \ldots))),$$

$$)) \ldots)))\,\phi$$

Como as cláusulas geradas a partir de $\mathcal{N}(r)$ contêm todas as cláusulas geradas a partir

de $(\beta_m \vee \beta_{m-1} \vee \ldots \beta_1 \vee \omega_n \vee \omega_{n-1} \vee \ldots \omega_1)\,\phi$

temos que:

$$C_R \in \mathrm{CLAU}\,((\beta_m \vee \beta_{m-1} \vee \ldots \beta_1 \vee \omega_n \vee \omega_{n-1} \vee \ldots \omega_1)\,\phi \;\subset\; \mathrm{CLAU}(\mathcal{N}(r))$$

Portanto:

$$C_R \in \mathrm{CLAU}(\mathcal{N}(r))$$

Finalmente, pelo Lema 6.2, $\text{CLAU}(\mathcal{N}(r)) = \text{CLAU}(r)$, temos:

$$C_R \in \text{CLAU}(r)$$

❑

Utilizando o lema anterior a cada passo da prova, vamos mostrar a completude do MSEL como decorrência da completude da Resolução.

Teorema 6.3

Seja Γ uma base de conhecimento composta por um conjunto de fórmulas e sejam $\vdash_R$ a dedução utilizando a Resolução e $\vdash$ a dedução utilizando o MSEL.

$$\text{Se } \text{CLAU}(\Gamma) \vdash_R \perp \text{, então } \Gamma \vdash \perp$$

Prova:

Seja $\Gamma = \Gamma_1$ e $\text{CLAU}(\Gamma_{i+1}) = \text{CLAU}(\Gamma_i) \cup C_i$ então, se $\text{CLAU}(\Gamma) \vdash_R \perp$ é uma prova utilizando a Resolução, temos:

$\text{CLAU}(\Gamma_1) \vdash_R C_1$,

$\text{CLAU}(\Gamma_2) \vdash_R C_2$,

... ,

$\text{CLAU}(\Gamma_i) \vdash_R C_i$,

... ,

$\text{CLAU}(\Gamma_n) \vdash_R \perp$

Pelo lema anterior, para cada passo da prova:

Se $\text{CLAU}(\Gamma_i) \vdash_R C_i$, então $\exists r$, tal que $\Gamma_i \vdash r$ e $C_i \in \text{CLAU}(r)$

Logo é possível deduzir

$\Gamma_1 \vdash r_1$, onde $C_i \in$ CLAU(r_1)

$\Gamma_2 \vdash r_2$, onde $C_2 \in$ CLAU(r_2),

... ,

$\Gamma_i \vdash r_i$, onde $C_i \in$ CLAU(r_i),

... ,

$\Gamma_n \vdash r_n$, onde $\perp \in$ CLAU(r_n)

Se $\perp \in$ CLAU(r_n) então, pelas regras de simplificação do MSEL, simplify(r_n) = $\perp$, logo:

$$\Gamma_n \vdash \perp$$

❑

Em conseqüência do teorema acima, temos como corolário que se a Resolução acrescida de uma estratégia é completa, então é possível construir uma estratégia completa no MSEL, análoga à da Resolução, onde, para cada passo da prova, $\Gamma_i \vdash r_i$, somente quando CLAU(Γ_i)$\vdash_R C_i$.

7. CONCLUSÃO

Este capítulo sumariza as conclusões e principais contribuições deste trabalho, e aponta algumas alternativas relacionadas com trabalhos futuros.

Nossa primeira conclusão é de que é possível tornar o método de Tableaux equivalente à Resolução.

As origens deste trabalho estão exatamente na comparação entre o método de Tableaux e Resolução. Procurávamos dotar o método de Tableaux de heurísticas comparáveis às da Resolução. Segundo nossos, critérios de comparação, além da ausência de heurísticas, a recuperação da falha no método de Tableaux também é menos eficiente. Na resolução, a falha é reconhecida a cada literal isolado que não conseguimos resolver, enquanto no Tableau a falha ocorre somente quando atingimos o limite do tamanho da prova.

A questão das heurísticas pode ser resolvida, se utilizarmos a polaridade dos literais para selecionar quais fórmulas desenvolver. Se, num ramo, temos a presença de uma fórmula α, onde ocorre um literal L_1 com polaridade positiva, vamos procurar desenvolver fórmulas onde L_1 ocorra com polaridade negativa. A presença dos dois literais complementares fecharia o ramo.

A falha é reconhecida quando L_1 é um literal isolado, quando não existe um literal complementar capaz de unificar com L_1. Todavia, a falha nesse caso não é uma falha de toda a fórmula α. Como α é uma fórmula genérica, e não uma cláusula, um outro literal L_2 de α poderia ser selecionado para a tentativa de fechar o ramo.

A recuperação da falha consiste em selecionar-se L_2 eficientemente. Para tanto, basta escolher um dos literais de $\alpha_1 = \alpha\ (L_1//T)$ – o resultado da substituição com ajuste de polaridade de L_1 pela constante lógica T em α, efetuadas as simplificações de praxe. Se α_1 = T, todos os literais de α falharam, consequentemente, deve ser feito o backtracking.

O MSEL é uma evolução desses conceitos. O MSEL, nossa contribuição principal, é um método refutacional baseado no fato de que numa fórmula contraditória necessariamente existem dois literais de polaridade complementar.

A fórmula é contraditória se, após uma conveniente eliminação de todos os pares de literais complementares, encontramos a fórmula vazia.

Comparamos o passo de dedução do MSEL com o da Resolução, e verificamos que, quando aplicado a cláusulas, o MSEL gera o mesmo resolvente. Se restringirmos nossas fórmulas a cláusulas de Horn, nossa heurística priorizará a cabeça das regras, funcionando analogamente a SLD-resolution.

Além disso vimos que para a aplicação do MSEL não necessitamos traduzir as fórmulas apresentadas pelo usuário para a forma clausal. Assim sendo, são evitadas algumas características indesejáveis da Resolução decorrentes do processo de tradução das fórmulas para a forma clausal.

Comparado com o método de Tableaux, o MSEL tem como vantagem dispor de mecanismos que permitam a utilização de bases de conhecimento. O problema da complexidade é minorado direcionando-se convenientemente a pesquisa na base de conhecimento, de acordo com as heurísticas disponíveis para a resolução, acrescidas das que apresentamos.

Sintetizando, podemos ressaltar os seguintes pontos positivos do MSEL:

- Preserva a intenção dos usuários e possibilita ganhos na estratégia de prova
- Evita redundância na passagem para forma clausal conjuntiva
- Facilita a Explicação
- Tem custo menor durante o pré-processamento. O cálculo das polaridades é mais simples que a conversão para fórmula clausal

Por outro lado, como aspecto negativo do MSEL, temos que o custo de simplificação das fórmulas durante a dedução pode ser mais alto do que os correspondentes passos utilizando Resolução.

O problema da Prova de Teoremas é inerentemente um problema de busca, onde a quantidade exponencial de caminhos possíveis torna imperiosamente necessária a utilização de heurísticas.

Assim sendo, os ganhos possibilitados com melhores estratégias, em particular com estratégias que levem em consideração a intenção do usuário, tendem a ser de ordem muito superior ao pequeno custo adicional em cada passo do processamento.

Além disso, respeitando-se a intenção do usuário, tende-se a gerar provas mais fáceis de explicar.

As estratégias apresentadas são nossa terceira contribuição.

Utilizar a direção indicada nas implicações, para direcionar a prova, parece-nos uma heurística simples e muito natural. Ao traduzir as fórmulas para cláusulas, a resolução perde essa informação essencial. Da mesma forma, o método de Tableaux também não recorre a esse direcionamento. Nossa segunda estratégia, de preferir literais com menor nível, é uma generalização da estratégia de preferir fatos a regras e é essencial para escolha de qual fórmula será a hospedeira e qual será visitante. Uma das virtudes do MSEL é permitir a utilização dessas heurísticas.

As estratégias apresentadas podem ser adaptadas para o método de Tableaux ou para Resolução desde que sejam acrescidas a esses métodos estruturas de dados auxiliares capazes de armazenar as informações necessárias à aplicação dessas heurísticas.

De um ponto de vista computacional, a disponibilidade de heurísticas é muito mais relevante que a decisão sobre qual método de prova será utilizado.

Existem algumas linhas de desenvolvimento posterior para este trabalho. A primeira diz respeito às diversas variantes do MSEL, que foram testadas, mas não utilizadas nesta tese. Optamos pela variante mais simples que possibilitasse a utilização das heurísticas desejadas. Outras variantes poderão ser melhor investigadas.

Neste trabalho, apresentamos apenas estratégias de ordenação. Essa opção facilita a prova de completude, uma vez que indicamos apenas uma preferência sem podar nenhum caminho de prova. O MSEL permite a utilização de heurísticas de refinamento que não foram exploradas neste trabalho.

Investigar como o MSEL pode ser estendido para outras lógicas também é uma questão que se inclui em nosso raio de interesses.

De uma maneira geral, acreditamos que a questão da eficiência dos métodos de prova é função da estruturação do conhecimento. Estudos relacionados com a estruturação do conhecimento são relevantes para quaisquer métodos de prova.

Enfim, devido à maior disponibilidade de memória, novos algoritmos de prova mais rápidos, mas que consomem mais memória, podem ser desenvolvidos.

8. BIBLIOGRAFIA

AVRON,ARNON, 1993- Gentzen-Type Systems , Resolution and Tableaux - *Journal of Automated Reasoning* 10, p.256-281, (1993)- Kluwer Academic Publishers, Holanda

BASIN,D.,FRONHÖFER, D., HÄNDLE, R., POSSEGA , J. AND SCHWIND,C, 1993.: *2nd Workshop on Theorem Proving with Analitic Tableaux and Related Methods.* Technical Report 213, Max-Planck Institut für Informatik, Saarbrücken, Germany, 1993.

BHATTA, K.S.H.S.R. AND KARNICK,HARISH ,1991- "A Resolution Rule for well-formed Fórmula" - *Theoretical Computer Science*, 81 p.223-235 (1991) - Elsevier Science Publishers B.V.

BECKERT, BERNHARD AND POSSEGA, JOACHIM ,1995 –" LeanT^{A}P: Lean Tableau-based Deduction" - *Journal of Automated Reasoning* 15, p.339-358, (1995)- Kluwer Academic Publishers, Holanda.

BETH,E.W,1956- Semantic Construction of Intuitionistic Logic, N.V.Noord-Hollandsche Uitgevers Maatschappij , Amsterdam, 1956

BETH, E.W. ,1964 - *The Foundations of Mathematics* - Harper & Row Publishers, New York, 1964.

BIBEL,W. AND EDER,E. ,1993 – "Methods and Calculli for Deduction - In C.J. Hogger Dov Gabbay and J.A.Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming - Vol 1 - Logical Foundations* - Oxford Science Publications, Oxford,1993.

BLEDSOE, W.W,1971 – Splitting and Reduction Heuristics in Automatic Theorem Proving – *Artificial Inteligence* , Vol. 2, North_Holland Publising Company,1971

BAUMGARTNER, PETER, 1996 - Theory Reasoning in Connection Calculi and the Linearizing Completion Approach – Ph. D. Thesis – Universität Klobentz – Landau - 1996

BAUMGARTNER, PETER; FURBACH ULRICH, 1994 – "Consolution as Framework for Comparing Calculi" - *J. SymbolicComputation* , Vol. 11,1994

BAUMGARTNER, PETER; HAHNLE, REINER; POSSEGA,JOACHIM (eds) , 1995- *Theorem Proving With Analytic Tableaux and Related Methods* : 4th International Workshop, Tableaux '95 Schloss Rheinfels, St. Goar, Germany May 7-10,

BAUMGARTNER, PETER; FURBACH ULRICH, 1997 – "Refinements for Restart Model Elimination" – in *Proceedings of FTP97 International Workshop on First order Theorem Proving* - Schloss Hagenberg, Austria - October 27 - 28, 1997

BIERWALD,CARSTEN; THOMAS KÄUFÍ- 1997,"Tableaux ProverTatzelwurm: Hyper-Links and UR-Resolution" - – in *Proceedings of FTP97 International Workshop on First order Theorem Proving* - Schloss Hagenberg, Austria - October 27 - 28, 1997

BRODA,K. , GABBAY, D.M. , KRIWACZEK,F. ,1985 - *A Goal Directed Theorem Prover for Predicate Logic Based on Conjuntions ans Implications* - Imperial College of Science and Technology - Draft Report - Londres,1985

BRODA,K., DÁGOSTINO,M.,GORÉ,R. JOHNSON,R. AND REEVES,S -. 1994: ..: *3rd Workshop on Theorem Proving with Analitic Tableaux and Related Methods.* Technical Report TR-94/5, Imperial College London, Departament of Computing, London, England, 1994.

CARVALHO, ROBERTO LINS , 1995– *SAFO - Documentação Técnica*, 1995 (Documento Reservado)

CARVALHO, ROBERTO LINS, 1988 – *Fundamentos Teóricos para Representação do Conhecimento*, LNCC, 1988

CARVALHO, ROBERTO LINS, 1974 – *Some Results in Automatic Theorem Proving with aplications in Elementary Set Theoryand Topology*- Ph.D. Thesis – Toronto – Canada , 1974

CARVALHO, ROBERTO LINS et alli, 1987 – *Engenharia do Conhecimento*, 2ª EBAI, Tandil- Argentina,1987

CASANOVA,MARCO A. , GIORNO, FERNANDO A.C. , FURTADO, ANTÔNIO L. , 1987 - *Programação em Lógica e a Linguagem Prolog* - Editora Edgard Blücher Ltda., 1987.

CHANG,CHI-LIANG AND LEE, RICHARD CHAR-TUNG, 1973 - *Symbolic Logic and Mechanical Theorem Proving* - Academic Press, New York, 1973

COSTA, MARCOS MOTA DO CARMO, 1992 - *Introdução à Lógica Modal Aplicada a Computação* - VIII Escola de Computação, 1992

DAVIS,Martin,1993 - "First Order Logic" -In C.J. Hogger Dov Gabbay and J.A.Robinson, editors, -Handbook of Logic in Artificial Intelligence and Logic Programming - Vol 1 - Logical Foundations - Oxford Science Publications, Oxford,1993.

EISINGER,NORBERT AND OHBACH,HANS JÜRGEN ,1993- "Deduction Systems based on Resolution" - In C.J. Hogger Dov Gabbay and J.A.Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming - Vol 1 - Logical Foundations* - Oxford Science Publications, Oxford,1993.

ENDERDON, Herbert R. – *A Mathematical Introduction to Logic* - Academic Press Inc., 1972

FITTING, M.C. ,1988 – "First-Order Modal Tableaux" - *Journal of Automated Reasoning* 4, p.191-213, - Kluwer Academic Publishers, Holanda (1988).

FITTING,M.C. ,1990 - *First Order Logic and Automated Theorem Proving* - Springer-Verlag, New_York, 1990.

FITTING,MELVIN,1993 – "Basic Modal Logic" - In C.J. Hogger Dov Gabbay and J.A.Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming - Vol 1 - Logical Foundations* - Oxford Science Publications, Oxford,1993.

FRONHÖFER, B., HÄHNLE. R. AND KÄUFL,T ,1992: *Workshop on Theorem Proving with Analytic Tableaux and Related Methods.* Technical Report 8/92 , Universität Karlsruhe,Fakultät für Informatik, Alemanha, 1992 .

GABBAY,DOV M. AND KRIWACZEK,FRANK,1991 – "A family of Goal Directed Theorem Provers Based on Conjunction and Implication: Part 1 "- *Journal of Automated Reasoning* 7, p.511-536, (1991)- Kluwer Academic Publishers, Holanda

GABBAY, DOV M. , 1994 – "Classical vs Non-Classical Logics (the Universality of Classical Logic)." - In C.J. Hogger Dov Gabbay and J.A.Robinson, editors, - *Handbook of Logic in Artificial Intelligence and Logic Programming - Vol 2 - Deduction Methodologies* - Oxford Science Publications, Oxford,1994

GALLIER,J,H, 1987- *Logic and Computer Science - Foundations os Automatic Theorem Proving* - John Wiley & Sons, Inc., - New York (1987)

GALMICHE,DIDIER ,Ed., 1997- *Automated Reasoning With Analytic Tableaux and Related Methods* : International Conference, Tableaux 97, Pont-A-Mousson, France, May 1997

GENESERETH, MICHAEL R. AND NILSSON, NILS J. ,1987- *Logical Foundations of Artificial Inteligence* - Morgan Kaufmann Publishers,Inc. - California, 1987.

GÖRAN SUNDHOLM , 1983– "Systems of Deduction" - - In Dov Gabbay and F.Guenthner, editors, -*Handbook of Philosophical Logic - Vol 1 – p. 133-188* D. Reidel PublishingbCompany, 1983

GORDON, M. J.C., MELHAM, T, F, 1993 – *Introduction to Hol : a Theorem Proving Environment for Higher Order Logic.* Cambridge University Press, 1993

HÄNLE, REINER AND SCHMITT, PETER,1994 – "The Liberalized δ-Rule in Free Variable Semantic Tableaux" - *Journal of Automated Reasoning* 13, p.211-221, (1994)- Kluwer Academic Publishers, Holanda.

HARRISON, JOHN, 1997 – "First Order Logic in Practice" – in Proceedings of FTP97 International Workshop on First order Theorem Proving Schloss Hagenberg, Austria - October 27 - 28, 1997

KIRCHNER, C. , KIRCHNER, H. , Ed.,1998 – *15th International Conference on Automated Deduction - Proceedings*, Lindau, Alemanha, Lecture Notes in Artificial Intelligence, 1421, Springer Verlag , julho,1998.

KLEENE, Stephen C., 1964 – *Introduction to Methamathematics*, 5ªediçao- D. Van Nostraand Company, 1964

KLEENE, Stephen C., 1967 – *Mathematical Logic* - John Wiley & Sons Inc., 1967

KREISEL G. , KRIVINE J.L., 1967 – *Elements of Mathematical Logic* – North Holland Publishing Company, Amsterdam , 1967

LI, D. 1997 – *Unification Algorithms for Eliminating and Introducing Quantifiers in Natural Deduction Automated Theorem Proving* – Journal of Automated Reasoning 18(1), pp 105.134, 1995

LOVELAND, Donald W. , 1978– *Automatic Theorem Proving: A Logical Basis* - North-Holland Publishing Company, 1978

MASSACCI, FABIO, 1997- "Simplification with Renaming: A General Proof Technique for Tableau and Sequent-Based Provers" - http://hermes.dis.uniroma1.it /~massacci /papers

MACKENZIE, DONALD, 1995 – The Automation of Proof: A Historical and Sociological Exploration – Annals of the History of Computing, IEEE, 1995, http://dream.dai.ed.ac.uk/papers/donald/donald.html

MCCUNE, W.W. AND WOS, LARRY,1997 – Otter. The CADE-13 Competition Incarnations – Journal of Automated Reasoning, 18(2), pp. 211-220 , 1997.

MENDES SANTOS,SUELI AND MILLAN, MARILIA ROSA, 1972 -Automatic Proofs for Theorems on Predicate Calculus,1972

MENEZES, C ; TAVARES O., 1988 –*Safo : Um ambiente para processamento do Conhecimento* – Anais do V Simpósio Brasileiro de Inteligência Artificial, 1988.

MIGLIOLI, P, , Editor, 1996 - *Theorem Proving With Analytic Tableaux and Related Methods* : 5th International Workshop, Tableaux '96, Terrasini, Palermo, Italy, 15-17th, 1996

MONDATORI , MARCO, 1995– "Efficient Inverse Tableaux" – Journal of te IGPL,Vol 3, n 6, pp.939-953, 1995

MURRAY, NEIL V. AND ROSENTHAL, ERIK, 1988 – "An Implementation of Dissolution-Based System Employng Theory Links" - *CADE Proceedings-Lecture Notes in Artificial Inteligence* p.658-674- Springer-Verlag, Alemanha.

MURRAY, NEIL V. , 1982 – "Completely Non-Clausal Theorem Proving" - *Artificial Inteligence* 18 (1982) 67-85 - North-Holland

NIEMELÄ, I. ,1988 – "Decision procedure for autoepistemic logic" - *In Proceedings of the Ninth International Conference on Automated Deduction* , p. 675-684, Argonne,ILL,1988.

NILSSON,NILS J.,1971 – *Problem Solving Methods in Artificial Inteligence* – McGral-Hill Book Company ,1971

NILSSON, NILS J. – 1988 – *Principles d'Intelligence Artificielle*, CEPADUES-EDITIONS, 1988

OLIVEIRA, CLAUDIA M.G.M ,1995- *An Architeture for Labeled Theorem Proving - Theoretical Aspects and Implementation* - Ph.D. Thesis, Imperial College of Science, Technology and Medicine - Londres ,1995.

PATTERSON,DAN W. , 1990- *Introduction to Artificial Inteligence and Expert Systems*, Prentice-Hall International,Inc.,1990.

PAULSON, L. , 1994 – *Isabelle, A generic Theorem Prover* – Lecture Notes in computer Sciense, Springer Verlag, 1994

PITT, JEREMY , CUNNINGHAM, JIM , 1996– "Theorem Proving and Model Building with Calculus KE" – *Journal of the IGPL* , Vol 4., n 1, pp.129-150 ,1996

PLAISTED, DAVID – "Ordered Semantic Hyper-Linking"- Report MPI-I-94-235 -Max-Planck-Institut für Informatic - 1994

REIF, WOLFGAN ; SCHELLHORN, GERHARD, 1997 – "Theorem Proving in Large Theories" – in *Proceedings of FTP97 International Workshop on First order Theorem Proving* - Schloss Hagenberg, Austria - October 27 - 28, 1997

RICH,ELAINE E KNIGHT,KEVIN, 1994- *Inteligência Artificial* - 2a. edição - Makron Books do Brasil, 1994.

RISCH,VICENT AND SCHWIND,CAMILLA, 1994 – "Tableau-Based Characterization and Theorem Proving for Default Logic" - *Journal of Automated Reasoning* 13, p.221-242, (1994)- Kluwer Academic Publishers, Holanda

ROBINSON, .J. A , 1965– "A Machine-Oriented Logic Based on Resolution Principle" - *Journal of Association for Computing Machinery* - Vol. 12, n 1, pp.23-41, (Janeiro,1965)

RUSSEL, STUART ; NORVIG, PETER, 1995 -*Artificial Intelligence : A Modern Approach (Prentice Hall Series in Artificial Intelligence);* Prentice Hall Press; 1995

RYAN, MARK D. , 1992 - *Ordered Presentation Theories* - Phd. Thesis - Department of Computing - Imperial College of Science , Technology and Medicine - Londres ,1992.

SMULLYAN, R.M. , 1968 – *First Order Logic*, Spring-Verlag, Berlin, 1968

SWART, HARRIE, Ed., 1998 - *Automated Reasoning With Analytic Tableaux and Related Methods : International Conference, Tableaux'98*, Oisterwijk, the Netherlands, 1998

TAMMET, T, 1997 – *Gandalf* – Journal of Automated Reasoning vol 18, n 2, (1997) , p. 199-204.

UNGAR, A.M. , 1992 – "Normalization, Cut-elimimation and Theory of Proofs" - *CSLI Lecture Notes n° 28* - United States, 1992

VIEIRA, N. J ,1987 – *Máquina de Inferência para Processamento de Conhecimento*- PUC/RJ – DI , Tese de Doutorado , 1987

VIEIRA, N. J , 1985– *SAFO – Manual do Usuário*,Rio, PUC/RJ – Departamento de INformática, 1985 (circulação Interna)

WANG,HAO ,1984 – "Computer Theorem Proving and Artificial Intelligence" – *Contemporary Mathematics* - Vol. 29 , - American Methematical Society, 1984

WALLEN , LINCOLN A. , 1990 - *Automated Deduction in Non Classical Logics* - The Mit Press, Massachusetts,1990.

WINSTON, HENRY PATRICK, 1987 - *Inteligência Artificial* -Livros Técnicos e Científicos Editora Ltda. - 1987.

WOS, LARRY AND VEROFF,ROBERT ,1994- "Logical basis for the Automation of Reasoning : Case studies" - In C.J. Hogger Dov Gabbay and J.A.Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming - Vol 2 - Deduction Methodologies* - Oxford Science Publications, Oxford,1994

WOS, LAWRENCE, ROBINSON, GEORGE ;CARSON,DANIELF., 1965 - "Efficiency ans Completeness of the Set of Suport Strategy in Theorem Proving" – Journal of Association for Computing Machinery - Vol. 12, n 4, pp.536-541, (Outubro,1965)