3
Método Horizontal

Este método unifica ramos em uma prova e, por meio de introduções de uma nova regra de "atenuação", constrói uma prova menor da mesma conclusão.

Apresentaremos na primeira seção deste capítulo a definição de unificador, o algoritmo de unificação e exemplo de aplicação do algoritmo em formulas proposicionais.

Na segunda seção apresentaremos o método e na terceira formalizaremos o método utilizando cálculo de seqüentes.

3.1
Definições

Definição 3.1 *Uma substituição θ é denominada um* unificador *para o conjunto $\{E_1, \ldots, E_k\}$ se e somente se $E_1\theta = E_2\theta = \cdots = E_k\theta$. O conjunto $\{E_1, \ldots, E_k\}$ é dito* unificável *se existe um unificador para ele.*

Definição 3.2 *O* par em desacordo *de um conjunto não vazio de expressões W é obtido localizando os primeiros símbolos (contando da esquerda) no qual nem todos as expressões em W tem exatamente os mesmos símbolos, e então extraindo de cada expressão em W a sub-expressão que começa com o símbolo ocupando esta posição. O conjunto desta respectivas sub-expressões é o* conjunto em desacordo *de W.*

Exemplo 3.3 *Se W é $\{\rightarrow\ (\rightarrow\ (\underline{A_2}, A_3), \rightarrow\ (\rightarrow\ (A_1, A_2), \rightarrow\ (A_1, A_3))); \rightarrow\ (\rightarrow\ (\underline{A_1}, A_3), \rightarrow\ (\rightarrow\ (A_4, A_1), \rightarrow\ (A_4, A_3)))\}$, então as primeiras posições na qual nem todas fórmulas em W são exatamente as mesmas é a quinta, uma vez que todas fórmulas possuem os quatro primeiros símbolos iguais $\rightarrow (\rightarrow ($. Então o conjunto em desacordo consiste das respectivas sub-expressões (termos sublinhados) que começam na posição cinco, ou seja o conjunto $\{A_2, A_1\}$.*

Exemplo 3.4 *Considere o conjunto W do exemplo 3.3.*

1. *$\sigma_0 = \epsilon$ e $W_0 = W$. Desde que W_0 não é unitário, σ_0 não é o unificador de W.*

Algorithm 1 Algoritmo de Unificação adaptado por emparelhamento

1: $k = 0$, $S_k = S$, $P_k = \{\epsilon\}$ e $\sigma_k = \{\epsilon\}$.

2: Se S_k é unitário, substitua as variáveis originais (remanescentes) de S por novas variáveis aplicando $\alpha_k/(v_k, v_k)$ para cada variável original remanescente v_k e adicione $\alpha_k/(v_k, v_k)$ a σ_k; σ_k é o unificador de S. Caso contrário, se S_k não é unitário, encontre o par em desacordo D_k de S_k.

3: Se existirem elementos v_k e t_k em D_k tal que v_k seja uma variável que não ocorre t_k, vá para o passo 4. Caso contrário, pare; S não é unificável.

4: Se $D_k \notin S_k$, construa S_{k+1} substituindo as ocorrências de D_k em S_k por α_k, onde α_k é uma variável que não está presente nem S e nem S_k. Caso contrário, construa S_{k+1} substituindo as ocorrências de D_k em S_k pelo α_k previamente associado. Faça $S_{k+1} = S_k \cup D_k$

5: Faça $k = k + 1$ e vá para o passo 2.

2. *O par em desacordo $D_0 = (A_2, A_1)$. Em D_0, existe uma variável $v_0 = A_2$ que não ocorre em $t_0 = A_1$.*

3. *$D_0 \notin \sigma_0$. Faça $\sigma_1 = \alpha_0/(A_2, A_1)$.*

$$\begin{aligned} S_1 = \{ &\to (\to (\alpha_0, A_3), \to (\to (A_1, A_2), \to (A_1, A_3))), \\ &\to (\to (\alpha_0, A_3), \to (\to (A_4, A_1), \to (A_4, A_3)))\} \end{aligned}$$

e $P_1 = \{(A_2, A_1)\}$.

4. *S_1 não é unitário e o par em desacordo de S_1 é:*

$$D_1 = (A_1, A_4)$$

5. *De D_1, temos que $v_1 = A_2$ e $t_1 = A_4$.*

6. *Como $D_1 \notin S_1$. Seja $\sigma_2 = \{\alpha_0/(A_2, A)_1), \alpha_1/(A_1, A_4)\}$.*

$$\begin{aligned} S_1 = \{ &\to (\to (\alpha_0, A_3), \to (\to (\alpha_1, A_2), \to (A_1, A_3))), \\ &\to (\to (\alpha_0, A_3), \to (\to (\alpha_1, A_1), \to (A_4, A_3)))\} \end{aligned}$$

e $P_2 = \{(A_2, A_1), (A_1, A_4)\}$.

7. *Uma vez que os próximos pares em desacordo já possuem um α_i previamente associado, geramos após dois passos*

$$\begin{aligned} \sigma_4 = &\{\alpha_0/(A_2, A)_1), \alpha_1/(A_1, A_4)\} \\ S_4 = \{ &\to (\to (\alpha_0, A_3), \to (\to (\alpha_1, \alpha_0), \to (\alpha_1, A_3))), \\ &\to (\to (\alpha_0, A_3), \to (\to (\alpha_1, \alpha_0), \to (\alpha_1, A_3)))\} \end{aligned}$$

e $P_4 = \{(A_2, A_1), (A_1, A_4)\}$.

S_4 *é unitário e uma variável original que permanece é* A_3*. Aplicando*

$$\alpha_2/(A_3, A_3)$$

geramos

$$\sigma_5 = \{\alpha_0/(A_2, A)_1), \alpha_1/(A_1, A_4), \alpha_2/(A_3, A_3)\}$$

e

$$S_5 = \{\rightarrow (\rightarrow (\alpha_0, \alpha_2), \rightarrow (\rightarrow (\alpha_1, \alpha_0), \rightarrow (\alpha_1, \alpha_2)))\}$$

que contém a fórmula resultante do processo de unificação.

Teorema 3.5 *Se* $S = \{\gamma_1, \gamma_2\}$ *é um conjunto com um par de fórmulas proposicionais então o algoritmo 1 sempre pára em tempo* $O(n)$*, onde* $n = max\{|\gamma_i| \ ; \ \gamma_i \in S, i = 1, 2\}$.

Prova.

Primeiramente, provaremos que o algoritmo pára.

Suponha por absurdo que o algoritmo não pare.

Desta forma, seria gerada uma seqüência infinita $S\sigma_0, S\sigma_1, S\sigma_2 \ldots$ de conjuntos de fórmulas, finitos e não vazios com a propriedade que cada novo conjunto contém uma variável original (remanescente) a menos que seu predecessor (por exemplo, $S\sigma_k$ contém v_k enquanto $S\sigma_{k+1}$ não contém).

Isto é impossível, pois S contém uma quantidade finita de variáveis.

Para provar a estimativa de tempo consumido pelo algoritmo, basta observar que o algoritmo caminha simultaneamente por γ_1 e γ_2 comparando os símbolos de cada um das fórmulas, substituindo cada símbolo por uma nova variável e armazenando informações em dois conjuntos, P_k e σ_k. Como o tempo para realizar estas operações pode ser considerado $O(1)$, temos que o algoritmo consumirá $n.O(1) = O(n)$. ■

3.2 O Método

Nosso método para reduzir o tamanho de uma prova durante o processo de construção consiste em substituir duas fórmulas similares pela fórmula resultante obtida pelo algoritmo 1.

Começamos construindo uma prova normal de σ de baixo para cima usando o conjunto de premissas Σ. Se durante a construção da prova encontramos duas fórmulas similares, no sentido apresentado no algoritmo 1, construímos uma prova da fórmula resultante do algoritmo de unificação em vez de construir dois novos ramos para cada uma das fórmulas similares.

Uma representação gráfica do método é apresentada na figura 3.1.

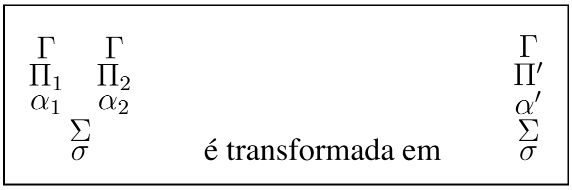


Figura 3.1: Representação gráfica do método

Antes de apresentar o método em detalhes, vejamos como seria a sua aplicação em um exemplo.

Exemplo 3.6 *Considere a seguinte prova de uma adaptação do numeral de Church* $(A_1 \to A_2) \to ((A_2 \to A_3) \to ((A_4 \to A_1) \to (A_4 \to A_3)))$ *(maiores detalhes sobre a adaptação do numeral de Church são apresentados na apêndice C).*

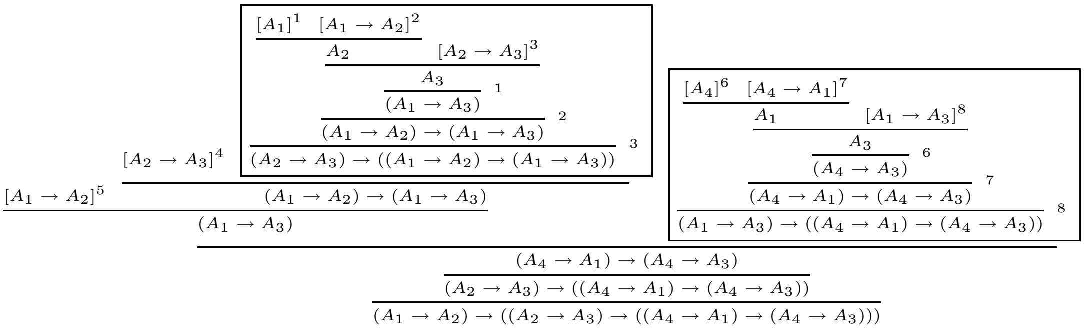


Figura 3.2: Aplicação do método vertical

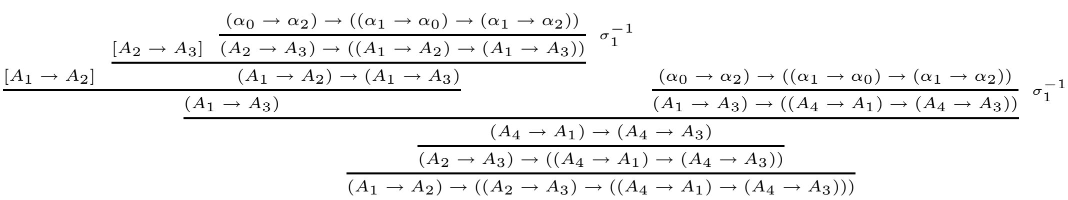
Temos, pelo exemplo 3.4 que as fórmulas no final das áreas marcadas na figura 3.2 são unificáveis com unificador

$$\sigma = \{\alpha_0/(A_2, A)_1), \alpha_1/(A_1, A_4), \alpha_2/(A_3, A_3)\}$$

e fórmula resultante

$$(\alpha_0 \to \alpha_2) \to ((\alpha_1 \to \alpha_0) \to (\alpha_1 \to \alpha_2)).$$

O que faremos, então, será substituir as duas derivações que estão marcadas na figura por uma nova derivação.



onde a derivação de $(\alpha_0 \to \alpha_2) \to ((\alpha_1 \to \alpha_0) \to (\alpha_1 \to \alpha_2))$ *é apresentada abaixo.*

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\alpha_1 \quad (\alpha_1 \to \alpha_0)}{\alpha_0} \qquad (\alpha_0 \to \alpha_2)}{\alpha_2}}{(\alpha_1 \to \alpha_2)}}{(\alpha_1 \to \alpha_0) \to (\alpha_1 \to \alpha_2)}}{(\alpha_0 \to \alpha_2) \to ((\alpha_1 \to \alpha_0) \to (\alpha_1 \to \alpha_2))}$$

Como dito no capítulo 2, para que o sistema de dedução natural se encaixe na definição de um sistema de Frege ele precisa ser visto como um grafo aciclico direto, ou seja, uma vez que uma fórmula é derivada em uma posição da prova se precisarmos novamente de tal fórmula não precisamos derivá-la outra vez. Desta forma, a derivação de $(\alpha_0 \to \alpha_2) \to ((\alpha_1 \to \alpha_0) \to (\alpha_1 \to \alpha_2))$ precisa ser realizada apenas uma vez, com isso reduzimos o tamanho da prova.

Uma possível apresentação de uma prova em dedução natural como um grafo acíclico direto é obtida se for utilizado o estilo devido a Fitch de apresentação de provas.

Exemplo 3.7 *Utilizando o estilo Fitch para exibir a prova do exemplo 3.6.*

1	α_1	*hipótese*
2	$\alpha_1 \rightarrow \alpha_0$	*hipótese*
3	α_0	*1,2 →E*
4	$\quad \alpha_0 \rightarrow \alpha_2$	*hipótese*
5	$\quad \alpha_2$	*2,3 →E*
6	$\alpha_1 \rightarrow \alpha_2$	*1, 5 →I*
7	$(\alpha_1 \rightarrow \alpha_0) \rightarrow (\alpha_1 \rightarrow \alpha_2)$	*2, 6 →I*
8	$(\alpha_0 \rightarrow \alpha_2) \rightarrow ((\alpha_1 \rightarrow \alpha_0) \rightarrow (\alpha_1 \rightarrow \alpha_2))$	*4, 7 →I*
9	$(A_2 \rightarrow A_3) \rightarrow ((A_1 \rightarrow A_2) \rightarrow (A_1 \rightarrow A_3))$	*8,* θ_1
10	$\quad (A_2 \rightarrow A_3)$	*hipótese*
11	$\quad (A_1 \rightarrow A_2) \rightarrow (A_1 \rightarrow A_3)$	*9, 10, →E*
12	$\qquad (A_1 \rightarrow A_2)$	*hipótese*
13	$\qquad (A_1 \rightarrow A_3)$	*11, 12 →E*
14	$(A_1 \rightarrow A_3) \rightarrow ((A_4 \rightarrow A_1) \rightarrow (A_4 \rightarrow A_3))$	*8,* θ_2
15	$(A_4 \rightarrow A_1) \rightarrow (A_4 \rightarrow A_3)$	*13, 14 →E*
16	$(A_2 \rightarrow A_3) \rightarrow ((A_4 \rightarrow A_1) \rightarrow (A_4 \rightarrow A_3))$	*10, 15 →I*

A prova é apresentada em 3 colunas. Na primeira coluna existe uma numeração que será utilizada como endereço da fórmula que é apresentada na segunda coluna. Na terceira coluna são informadas a origem da fórmula da segunda coluna. Por exemplo, a informação $1, 2, \rightarrow E$ *indica que a fórmula presente na segunda coluna foi obtida por* $\rightarrow E$ *usando as fórmulas das linhas 1 e 2 como premissas.*

Observe que a fórmula $(\alpha_0 \rightarrow \alpha_2) \rightarrow ((\alpha_1 \rightarrow \alpha_0) \rightarrow (\alpha_1 \rightarrow \alpha_2))$ *foi utilizada como premissa para deduzir as fórmulas* $(A_2 \rightarrow A_3) \rightarrow ((A_1 \rightarrow A_2) \rightarrow (A_1 \rightarrow A_3))$ *e* $(A_1 \rightarrow A_3) \rightarrow ((A_4 \rightarrow A_1) \rightarrow (A_4 \rightarrow A_3))$. *No entanto, só foi necessário realizar uma prova apenas de* $(\alpha_0 \rightarrow \alpha_2) \rightarrow ((\alpha_1 \rightarrow \alpha_0) \rightarrow (\alpha_1 \rightarrow \alpha_2))$.

3.3 Formalização do método

No que segue, formalizaremos os conceitos vistos nesta introdução de seção.

A apresentação de uma dedução como um grafo acíclico direto, em vez de apresentação como uma árvore, nos permite reduzir o tamanho da dedução, uma vez que apagamos da dedução subdeduções repetidas.

O que o nosso método propõe é eliminar não só os ramos idênticos, mas também os ramos similares. No nosso caso, ramos similares são aqueles em que as conclusões são unificáveis pelo algoritmo 1.

Voltando ao exemplo 3.6, observe que introduzimos uma nova regra, θE, que aplica uma substituição à fórmula que é sua premissa. Isto é,

$$\frac{\alpha}{\alpha'}\ \theta E$$

onde α' é a fórmula resultante do processo de unificação de α_1 e α_2 utilizando o algoritmo 1.

Ou seja, o ganho de redução do tamanho da prova é obtido por eliminar, de forma consistente, deduções de fórmulas similares.

Devemos observar certas condições para garantir a consistência neste novo sistema de prova. Primeiro, precisamos garantir que a nova fórmula obtida pelo processo de unificação não possui variáveis utilizadas previamente na prova e segundo não permitir a unificação de premissas que ainda serão utilizadas na prova.

O algoritmo de unificação gera uma fórmula que possui apenas variáveis novas, o que atende a primeira condição.

O segundo cuidado deve ser tomado durante a construção da prova, precisamos verificar se as fórmulas que foram empregadas na prova de α', serão descarregadas depois da conclusão. Isto é, devemos atentar para os ramos abertos da derivação. No exemplo a seguir tratamos de um caso com esse problema.

Exemplo 3.8 *Considere os dois trechos de prova a seguir.*

$$\dfrac{\dfrac{\dfrac{\dfrac{[A_1]^1 \quad [A_1 \to A_2]^2}{A_2} \qquad A_2 \to A_3}{A_3}}{(A_1 \to A_3)}\,{}^{1}}{(A_1 \to A_2) \to (A_1 \to A_3)}\,{}^{2} \qquad\qquad \dfrac{\dfrac{\dfrac{\dfrac{[A_4]^1 \quad [A_4 \to A_1]^2}{A_2} \qquad A_1 \to A_3}{A_3}}{(A_4 \to A_3)}\,{}^{1}}{(A_4 \to A_1) \to (A_4 \to A_3)}\,{}^{2}$$

Na segunda linha de cada dedução, contando de cima para baixo, aparecem as fórmulas $A_2 \to A_3$ e $A_1 \to A_3$ que serão descarregadas mais abaixo na prova.

Se aplicarmos o algoritmo 1 às fórmulas $(A_1 \to A_2) \to (A_1 \to A_3)$ e $(A_4 \to A_1) \to (A_4 \to A_3)$, obtermos um unificador $\theta = \{\alpha_0/(A_1, A_4), \alpha_1/(A_2, A_1), \alpha_2/(A_3, A_3)\}$ e fórmula resultante do processo de unificação $\alpha' = (\alpha_0 \to \alpha_1) \to (\alpha_0 \to \alpha_2)$.

Para realizar o processo de unificação de derivações de prova é preciso também aplicar θ as hipóteses não descarregadas. Fazendo isto, geramos o seguinte trecho de prova:

$$\dfrac{\dfrac{\dfrac{\dfrac{[\alpha_1]^1 \quad [\alpha_1 \to \alpha_1]^2}{\alpha_1} \qquad \alpha_1 \to \alpha_2}{\alpha_2}\,{\scriptstyle 1}}{(\alpha_0 \to \alpha_2)}\,{\scriptstyle 2}}{(\alpha_0 \to \alpha_1) \to (\alpha_0 \to \alpha_2)}$$

A fórmula $\alpha_1 \to \alpha_2$ ainda deve ser descarregada e convertida para sua fórmula original, pois no restante da prova tal fórmula não aparece.

Isto é, devemos descarregar a hipótese $(\alpha_1 \to \alpha_2)$ como premissa de $\to I$ aplicar $\theta_1 E$ e depois eliminar $(A_2 \to A_3)$ por $\to E$. Ou seja, fazemos uso de um "corte", como pode ser visto abaixo:

$$\dfrac{(A_2 \to A_3) \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[\alpha_1]^1 \quad [\alpha_1 \to \alpha_1]^2}{\alpha_1} \qquad [\alpha_1 \to \alpha_2]^*}{\alpha_2}\,{\scriptstyle 1}}{(\alpha_0 \to \alpha_2)}\,{\scriptstyle 2}}{(\alpha_0 \to \alpha_1) \to (\alpha_0 \to \alpha_2)}\,\to I^*}{(\alpha_1 \to \alpha_2) \to ((\alpha_0 \to \alpha_1) \to (\alpha_0 \to \alpha_2))}\,\theta_1 E}{(A_2 \to A_3) \to ((A_1 \to A_2) \to (A_1 \to A_3))}}{(A_1 \to A_2) \to (A_1 \to A_3)}\,\to E$$

O exemplo acima explica como devemos interpretar as traduções de hipóteses não descarregadas quando trabalhamos no sistema de dedução natural. Durante o processo de construção da prova, este "corte"se torna desnecessário.

Como a construção de provas em dedução natural com controle de hipóteses a serem utilizadas é quase o mesmo que construir provas em cálculo de seqüentes (vide (Hae90)) optamos por apresentar este formalismo em um cálculo de seqüentes simplificado, SEQ_0 devido a Schütte (Sch50).

3.3.1
O sistema SEQ_0

Apresentaremos nesta seção um sistema simplificado de cálculo de seqüentes, SEQ_0. Estes tipos de sistemas também são conhecidos por *seqüentes de um lado (One-Side sequent)* pois as informações de cada linha de uma derivação são colocadas apenas no lado direito da fórmula. Por este motivo se torna desnecessário o uso do símbolo separador $\vdash$ (ou $\Rightarrow$) na apresentação do cálculo.

Linguagem SEQ_0 (Forma de Schütte-Rasiowa-Sikorski)

Variáveis: $v_0, v_1, \ldots, v_i$

Literais: x ou $\neg x$.

Fórmulas: Construídas a partir dos literais usando dois conectivos, $\vee$ e $\wedge$.

Seqüentes: Conjunto finito de fórmulas (em particular qualquer fórmula α é um sequente $\{\alpha\}$). Em geral seqüentes são apresentados como listas de fórmulas, i.e., $\Gamma = \alpha_1, \alpha_2, \ldots, \alpha_k$ para significar $\{\alpha_1, \alpha_2, \ldots, \alpha_k\}$.

Regras

1. Axiomas $\Gamma, \alpha, \neg\alpha$
2. $\dfrac{\Gamma, \alpha, \beta}{\Gamma, \alpha \vee \beta}$ Regra $\vee$
3. $\dfrac{\Gamma, \alpha \qquad \Gamma, \beta}{\Gamma, \alpha \wedge \beta}$ Regra $\wedge$
4. $\dfrac{\Gamma}{\Gamma, \alpha}\ W$
5. $\dfrac{\Gamma, \alpha, \alpha}{\Gamma, \alpha}\ C$

$\neg\alpha$ é definido por indução usando as regras de De Morgan:

1. $\neg x := \neg x$
2. $\neg\neg x := x$
3. $\neg(\alpha \vee \beta) := \neg\alpha \wedge \neg\beta$
4. $\neg(\alpha \wedge \beta) := \neg\alpha \vee \neg\beta$

3.3.2
Dedução estruturada como árvore vs. dedução estruturada como circuito em SEQ_0

Dedução estruturada como árvore

Definição 3.9 *Seja* Γ *qualquer seqüente em* SEQ_0. *Uma* dedução de Γ estruturada como árvore *é uma árvore cujo vértice mais abaixo (=raiz) é* Γ, *e os vértices mais acima (=folhas) são axiomas, e exceto pelas folhas cada vértice é a conclusão de uma instância de uma regras de inferência em* SEQ_0 *cujas premissas estão logo acima da conclusão. O* peso *e a* altura *da dedução estruturada como árvore é o número total de vértices e o comprimento do caminho máximo, respectivamente.*

Dedução estruturada como circuitos

Definição 3.10 *Uma* dedução de Γ estruturada como circuito *é um circuito finito, i.e. um grafo acíclico direto, satisfazendo todas condições da definição anterior; o* peso *e a* altura *são definidos como no caso de dedução estruturada como árvore.*

Obviamente, toda dedução estruturada como árvore é uma dedução estruturada como circuito.

3.3.3 Regras auxiliares de inferência em SEQ_0

Corte

A regra do corte é uma inferência da forma

$$\frac{\Gamma, C \quad \Gamma, \neg C}{\Gamma}$$

como circuito

$$\begin{matrix} \Gamma, C \\ \Gamma, \neg C \end{matrix} \rightarrowtail \Gamma$$

para um fórmula arbitrária C.

Inversões válidas em SEQ_0.

Para qualquer seqüente em SEQ_0 definimos suas inversões válidas pelas seguintes cláusulas recursivas.

1. Γ, F e Γ, G são inversões válidas de $\Gamma, F \wedge G$.

2. Γ, F, G é inversão válida de $\Gamma, F \vee G$.

Atenuação por substituição em SEQ_0.

A *regra de atenuação por substituição*, WS, e a regra de *atenuação por substituição inversa*, WIS, em SEQ_0 são as seguintes inferências, respectivamente:

$$\frac{\Gamma}{\theta(\Gamma), \Sigma} \quad \text{e} \quad \frac{\Gamma}{\theta(\Gamma^-), \Sigma}$$

como circuito

$$\Gamma \rightarrowtail \theta(\Gamma), \Sigma \qquad \text{e} \qquad \Gamma \rightarrowtail \theta(\Gamma^-), \Sigma$$

para quaisquer Γ, Σ, inversão válida Γ^- de Γ em SEQ_0, e qualquer θ : Variáveis $\rightarrow$ Variáveis.

Teorema 3.11 *Para qualquer seqüente proposicional* $\Gamma = F_1, \ldots, F_k$*, a fórmula* $F_1 \vee \cdots \vee F_k$ *é válida em lógica proposicional se, e somente, se:*

1. Γ *tem uma dedução estruturada como árvore em* SEQ_0*,*

2. Γ *tem uma dedução estruturada como circuito em* SEQ_0*.*

Em particular, qualquer fórmula proposicional F *é válida em lógica proposicional se, e somente, se, existir uma dedução em* SEQ_0 *e se, e somente, se existir uma dedução estruturada como circuito em* SEQ_0*.*

Prova. O ítem 1 é bem conhecido. O item 2 é facilmente deduzido de 1. ■

Teorema 3.12 *A regra do corte é válida em* SEQ_0*. Isto é, se* Γ, C *e* $\Gamma, \neg C$ *são deduzidas em* SEQ_0 *então também o é* Γ*, tanto em como árvore quanto como circuito. Assim, adicionar corte a* SEQ_0 *não estende o conjunto de seqüentes que se podem deduzir, tanto como árvore quanto como circuito.*

Prova. A prova segue diretamente do teorema 3.11. ■

Teorema 3.13 *A regra atenuação por substituição inversa é válida em* SEQ_0*. Isto é, para quaisquer seqüentes* Γ *e* Σ*, qualquer substituição de variáveis* θ *e qualquer inversão válida* Γ^- *de* Γ*, se* Γ *é deduzida por uma dedução estruturada como árvore (ou por dedução estruturada como circuito) em* SEQ_0*, então também é deduzida como dedução estruturada como árvore (ou por dedução estruturada como circuito)* Γ^-*,*Σ*.*

Assim adicionar Corte e/ou adicionar WS e/ou adicionar WIS a SEQ_0 *não estende o conjunto de seqüentes que podem ser deduzidos em deduções estruturadas como árvores (ou em dedução estruturada como circuitos)*

Prova. Suponha que $\Gamma = F_1, \ldots, F_k$ é deduzida em SEQ_0. Assim, pelo teorema 3.11, $F_1 \vee \cdots \vee F_k$ é valida em lógica proposicional.

Agora, se $\Gamma^- = F_1^-, \ldots, F_l^-$ é uma inversão válida de Γ, então $F_1^- \vee \cdots \vee F_l^-$ também é uma fórmula válida. Ainda, como, por definição, a validade de fórmulas é preservada por atribuição de variáveis, temos que $\theta\left(F_1^-\right) \vee \cdots \vee \theta\left(F_l^-\right)$ também é válida, e obviamente também é válida qualquer atenuação $\theta\left(F_1^-\right) \vee \cdots \vee \theta\left(F_l^-\right) \vee G_1 \vee \cdots \vee G_m$, onde $\Sigma = G_1, \cdots, G_m$.

Disto e pelo teorema 3.11 obtemos a dedução de $\theta\left(\Gamma^-\right)$,$\Sigma$. ■

3.3.4 O Método em SEQ_0

Nosso método para reduzir o tamanho de uma prova durante o processo de construção da mesma, consiste em substituir duas fórmulas similares pela fórmula obtida pelo algoritmo 1.

Começamos construindo uma prova normal de α de baixo para cima usando Γ como premissas. Em um certo ponto da prova encontramos duas fórmulas unificáveis quando submetidas ao algoritmo 1. Construímos uma prova da fórmula obtida no final do algoritmo em vez de construir dois novos ramos de prova um para cada uma das fórmulas similares.

Uma representação gráfica do método horizontal em SEQ_0 pode ser vista na figura 3.3.

$$\begin{array}{c} \begin{array}{cc} D_1 & D_2 \\ \Gamma, \alpha_1 & \Gamma, \alpha_2 \end{array} \\ \vdots \\ \Gamma, \alpha \end{array} \quad \text{é transformada em} \quad \begin{array}{c} \dfrac{\begin{array}{c} D' \\ \Gamma, \alpha' \end{array}}{\Gamma, \alpha_1 \qquad \Gamma, \alpha_2}\,\text{WIS} \\ \vdots \\ \Gamma, \alpha \end{array}$$

i.e.

$$\left.\begin{array}{l} \stackrel{D_1}{\Rightarrow} \Gamma, \alpha_1 \\ \stackrel{D_2}{\Rightarrow} \Gamma, \alpha_2 \end{array}\right. \Rightarrow \Gamma, \alpha \quad \text{é transformada em} \quad \stackrel{D'}{\Rightarrow} \Gamma, \alpha' \rightarrowtail \begin{array}{c} \Gamma, \alpha_1 \\ \Gamma, \alpha_2 \end{array} \Rightarrow \Gamma, \alpha$$

Figura 3.3: Representação Gráfica do Método Horizontal

A regra WIS é a aplicação inversa do algoritmo 1, isto é, se duas fórmulas α_1 e α_2 são unificáveis quanto submetidas ao algoritmo 1 e a fórmula resultante do processo de unificação é α' então

$$\frac{\Gamma, \alpha'}{\Gamma, \alpha_1 \quad \Gamma, \alpha_2}\, {\scriptstyle WIS} \qquad \text{e} \qquad \frac{\Gamma, \alpha_1 \quad \Gamma, \alpha_2}{\Gamma, \alpha'}\, {\scriptstyle WS}$$

3.3.5 Exemplo de aplicação: Numerais de Church

Vejamos como o método se comporta em SEQ_0. Aplicamos o método ao exemplo 3.6, depois de convertê-lo a linguagem de SEQ_0.

Modus Ponens

Em SEQ_0, podemos inferir Γ, B de Γ, A e $\Gamma, \neg A \vee B$.

Usaremos MP para abreviar o esquema apresentado na figura 3.4.

$$\frac{\Gamma, A \quad \Gamma, \neg A \vee B}{\Gamma, B}$$

Figura 3.4: Modus Ponens

Exemplo 3.14 *Considere a prova de* $\sigma = (\neg A_1 \vee A_2) \vee (\neg(\neg A_4 \vee A_3) \vee (\neg(A_4 \vee A_1) \vee (\neg A_4 \vee A_3)))$.

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overset{\vdots}{(\neg A_1 \vee A_2)} \quad \dfrac{\overset{\vdots}{(\neg A_2 \vee A_3)} \quad \overset{\Pi_2}{F_2}}{\neg(\neg A_1 \vee A_2) \vee (\neg A_1 \vee A_3)}\,MP}{(\neg A_1 \vee A_3)}\,MP \quad \overset{\Pi_1}{F_1}}{\neg(A_4 \vee A_1) \vee (\neg A_4 \vee A_3)}\,MP}{\neg(\neg A_4 \vee A_3), \neg(A_4 \vee A_1) \vee (\neg A_4 \vee A_3)}}{\neg(\neg A_4 \vee A_3) \vee (\neg(A_4 \vee A_1) \vee (\neg A_4 \vee A_3))}}{\dfrac{(\neg A_1 \vee A_2), \neg(\neg A_4 \vee A_3) \vee (\neg(A_4 \vee A_1) \vee (\neg A_4 \vee A_3))}{(\neg A_1 \vee A_2) \vee (\neg(\neg A_4 \vee A_3) \vee (\neg(A_4 \vee A_1) \vee (\neg A_4 \vee A_3)))}}
$$

onde $F_1 = \neg(\neg A_1 \vee A_3) \vee (\neg(\neg A_4 \vee A_1) \vee (\neg A_4 \vee A_3))$ *e* $F_2 = \neg(\neg A_2 \vee A_3) \vee (\neg(\neg A_1 \vee A_2) \vee (\neg A_1 \vee A_3))$.

Π_1 *é a seguinte derivação:*

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A_4 \quad \overset{\vdots}{\neg A_4 \vee A_1}}{A_1}\,MP \quad \overset{\vdots}{\neg A_1 \vee A_3}}{A_3}\,MP}{\neg A_4, A_3}}{(\neg A_4 \vee A_3)}}{\neg(\neg A_4 \vee A_1), (\neg A_4 \vee A_3)}}{\neg(\neg A_4 \vee A_1) \vee (\neg A_4 \vee A_3))}}{\dfrac{\neg(\neg A_1 \vee A_3), (\neg(\neg A_4 \vee A_1) \vee (\neg A_4 \vee A_3))}{\neg(\neg A_1 \vee A_3) \vee (\neg(\neg A_4 \vee A_1) \vee (\neg A_4 \vee A_3))}}
$$

a derivação Π_2 *é similar a derivação de* Π_1

Aplicando o algoritmo 1 em $F_1 = \neg(\neg A_1 \vee A_3) \vee (\neg(\neg A_4 \vee A_1) \vee (\neg A_4 \vee A_3))$ e $F_2 = \neg(\neg A_2 \vee A_3) \vee (\neg(\neg A_1 \vee A_2) \vee (\neg A_1 \vee A_3))$ temos que as duas fórmulas são unificáveis.

O unificador é $\theta = \{\alpha_0/(\neg A_2, \neg A_1), \alpha_1/(\neg A_1, \neg A_4), \alpha_2/(\neg A_2, \neg A_1), \alpha_3/(A_3, A_3)\}$ e a fórmula resultante do processo de unificação é $F' = \neg(\alpha_0 \vee \alpha_3) \vee (\neg(\alpha_1 \vee \alpha_2) \vee (\alpha_1 \vee \alpha_3))$.

Seja Π' a derivação de F'. Temos que Π' é:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\neg\alpha_1 \quad \begin{matrix}\vdots\\ \alpha_1\vee\alpha_0\end{matrix}}{\alpha_0} \qquad \begin{matrix}\vdots\\ \neg\alpha_0\vee\alpha_3\end{matrix}}{\alpha_3}}{\alpha_1,\alpha_3}}{(\alpha_1\vee\alpha_3)}}{\neg(\alpha_1\vee\alpha_2),(\alpha_1\vee\alpha_3)}}{\neg(\alpha_1\vee\alpha_2)\vee(\alpha_1\vee\alpha_3)}}{\neg(\alpha_0\vee\alpha_3),\neg(\alpha_1\vee\alpha_2)\vee(\alpha_1\vee\alpha_3)}$$

a derivação é transformada em:

$$\Rightarrow F' \overset{WIS}{\rightarrowtail} \begin{array}{lll} F_1 \rightarrowtail & \dots \rightarrowtail & \\ & & \neg(\neg A_4 \vee A_1) \vee (\neg A_4 \vee A_3) \Rightarrow \gamma \\ F_2 & \rightarrowtail & \end{array}$$

3.3.6 Exemplo de aplicação: transitividade da implicação

Considere uma prova de $a_1 \rightarrow a_{2_k}$ a partir de um conjunto de hipóteses $\{a_i \rightarrow a_{i+1}, i = 1, 2, \dots k-1\}$. Isto é, a prova da tautologia $(a_1 \rightarrow a_2) \rightarrow (a_2 \rightarrow a_3) \dots (a_{2_{k-1}} \rightarrow a_{2_k}) \rightarrow (a_1 \rightarrow a_{2_k})$. Onde $2_k = \left. 2^{2^{2^{\cdot^{\cdot^{2}}}}} \right\} k$.

Uma prova normal desta fórmula possui $O(2_k)$ linhas. Por outro lado, se usamos a prova de $(A \rightarrow B) \rightarrow (B \rightarrow C) \rightarrow (A \rightarrow C)$ para tratar a transitividade de $\rightarrow$, e se partirmos em duas cada conclusão intermediária, isto é,

de $a_1 \rightarrow a_{2_{k-1}}$ e $a_{2_{k-1}} \rightarrow a_{2_k}$ derivar $a_1 \rightarrow a_{2_k}$

de $a_1 \rightarrow a_{2_{k-2}}$ e $a_{2_{k-2}} \rightarrow a_{2_{k-1}}$ derivar $a_1 \rightarrow a_{2_{k-1}}$

e o mesmo sendo aplicado as outras premissas de forma recursiva.

Teremos assim uma prova linear da fórmula apresentada inicialmente.

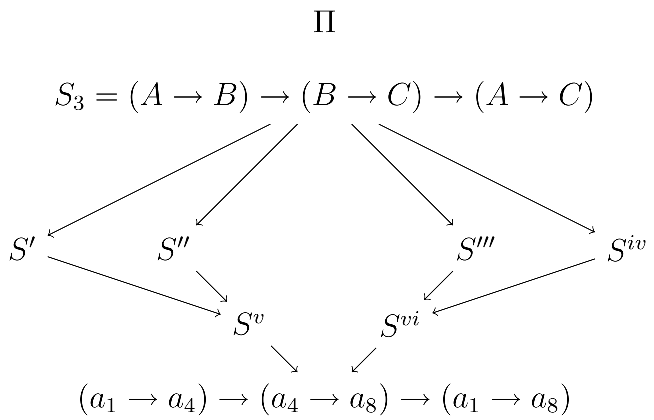


Figura 3.5: Esboço da redução da transitividade para $(a_1 \to a_4) \to (a_4 \to a_8) \to (a_1 \to a_8)$

Transitividade em SEQ_0

Seja Γ_{2_n} uma representação em SEQ_0 da fórmula acima.

$$\Gamma_{2_n} := \neg a_1, a_1 \wedge \neg a_2, a_2 \wedge \neg a_3, \ldots, a_{2_n - 1} \wedge \neg a_{2_n}, a_{2_n}$$

Uma prova de Γ_{2_n} em $SEQ_0 + WS$ teria o seguinte aspecto:

$$\frac{\Gamma_{2_{k-1}}, \Pi \quad \Delta, \Gamma^*_{2_{k-1}}}{\Gamma_{2_k}}$$

onde

$$\Gamma^*_{2_{k-1}} = \neg a_{2_{k-1}+1}, a_{2_{k-1}+1} \wedge \neg a_{2_{k-1}+2}, a_{2_{k-1}+2} \wedge \neg a_{2_{k-1}+3}, \ldots, a_{2_k - 1} \wedge \neg a_{2_k}, a_{2_k}.$$

Onde $\Gamma^*_{2_{k-1}} = \theta(\Gamma_{2_{k-1}})$ e θ é a seguinte substituição:

$$\theta(a_i) = a_{2_{k-1}+i}.$$

Aplicando o processo de substituição recursivamente em $\Gamma_{2_{k-1}}, \Gamma_{2_{k-2}}, \ldots$ obtemos a redução desejada também em $SEQ_0 + WS$.

3.3.7 Aplicação em PHP

O princípio das casas de pombos seria um bom candidato para se aplicar um processo de eliminação de ramos semelhantes na prova. Isto é, PHP seria um bom candidato para se trabalhar com $SEQ_0 + WS$.

Finger em (Fin05) apresenta uma prova polinomial do princípio da casas dos pombos como grafo acíclico direcionado (DAG) em Tableaux que podem ser facilmente traduzidas para $SEQ_0 + WS$ (sem corte). Contudo, a construção proposta por Finger faz uso de substituições não triviais (i.e substituições que exigem a intervenção do usuário).

Em suma, é possível produzir uma prova polinomial de PHP em SEQ_0+WS. Mas, analisando os trabalhos de Cook (Coo76) e Finger (Fin05) e as características de uma prova de PHP, temos fortes indícios para crer que as substituições necessárias para produzir uma prova curta dependeriam da intervenção do usuário.

Como a nossa intenção é apresentar métodos que não dependam da intervenção do usuário preferimos omitir a apresentação da tradução para $SEQ_0 + WS$ do exemplo apresentado por Finger.

3.4 Conclusão do método

O método apresentado nesta seção atende os requisitos propostos na introdução da presente tese. Senão, vejamos:

- O método pode ser aplicado durante o processo de construção da prova. Esta característica é de fundamental importância, uma vez que pretendemos utilizar este método em provadores automáticos de teorema.
- O método é de fácil implementação. Um algoritmo é fornecido para o caso em que a substituição seja um renomeamento de variáveis.
- O método é correto e completo. Como mostrado no teorema 3.13 a adição das regras WS e WIS não alteram o poder de SEQ_0 que é completo e correto.

Algumas melhorias naturais que podem ser acrescentadas ao método, que serão temas para trabalhos futuros, são:

- Apresentar uma maneira eficaz de buscar por pares de fórmulas semelhantes durante a construção do grafo de prova.

  Não abordamos na tese qual a melhor forma de buscar as possíveis fórmulas semelhantes na prova. Mostramos que, uma vez encontradas tais fórmulas semelhantes o processo de produzir o unificante consumiria um tempo linear.

  Apesar de não apresentarmos um processo de busca por fórmulas em um grafo acíclico, temos fortes indícios de que os recursos consumidos na busca seria compensado pelo redução de recursos necessários para a construção da prova.
- Estender o método para lógica de predicados.

  Todos os resultados apresentados para SEQ_0 são estendidos para SEQ, a versão predicativa de SEQ_0. Precisamos trabalhar num algoritmo de unificação que produza um substituição θ : *variáveis* $\rightarrow termos$.