2
Preliminares

Com intuito de fixar a notação, apresentaremos, neste capítulo, resultados e definições essenciais que serão utilizados ao longo da tese.

Para indicar identidade de duas expressões, usaremos $\equiv$.

$\mathbb{N}$ é utilizado para os números naturais, com zero incluído. Seguiremos as notações padrões da teoria dos conjuntos como $\in$, $\subset$.

Uma fórmula proposicional é representada por letras gregas minúsculas $\alpha, \beta, \gamma, \ldots$ (possivelmente com sub- ou supescrito ou $'$) sendo constituída de variáveis proposicionais $p, q, r, \ldots$ (possivelmente com sub- ou supescrito ou $'$), e de um conjunto de conectivos unários e binários $\neg, \wedge, \vee$ e $\rightarrow$.

O conjunto de variáveis proposicionais em uma fórmula α será denotado por $V(\alpha)$.

O tamanho de uma fórmula é definido pelo número total de símbolos que aparece na fórmula.

O tamanho de uma prova é definido pelo número total de símbolos na prova.

Definição 2.1 (Subfórmulas) *A noção de* subfórmula *que usaremos seguirá o sentido de Gentzen.*

Subfórmulas de α serão definidas por

(i) *α é uma subfórmula de α;*

(ii) *α é uma subfórmula de $\neg\alpha$;*

(iii) *Se $\gamma \square \beta$ é uma subfórmula de α então também são γ, β para $\square = \vee, \wedge, \rightarrow$.*

Nossa avaliação de complexidade de um método levará em conta a "taxa de crescimento" da função correspondente. Será usado o símbolo de ordem de magnitude O para lidar com este conceito.

Definição 2.2 *Considerando f e g como funções de $\mathbb{N}$ em $\mathbb{N}$. Então:*

$O(f)$ é o conjunto de funções g tais que para algum $r > 0$ e para todo n a menos de uma quantidade finita, $g(n) < r.f(n)$.

Usaremos como sistema de provas o sistema de Dedução Natural (ND) como em Prawitz (Pra65) seguindo a convenção de apresentar árvores de provas, colocando as premissas maiores das regras de eliminação no lado esquerdo.

2.1
Sistema de Dedução Natural

Definição 2.3 *(Sistemas de Dedução Natural)*

Um sistema de dedução natural *(ND) é determinado a partir de um conjunto de regras definidas sobre fórmulas de uma linguagem que representam a introdução e a eliminação dos operadores lógicos, e uma noção de dedução.*

As regras *são expressões da forma:*

$$\frac{S_1, S_2, \ldots, S_n}{S}$$

tal que $S_1, S_2, \ldots, S_n$ e S são fórmulas. Neste caso, dizemos que S é inferida a partir de $S_1, S_2, \ldots, S_n$ ou de $S_1, S_2, \ldots, S_n$ se infere S.

Em dedução natural, nenhum fórmula é considerada axiomaticamente válida, porém é permitido introduzir, a qualquer momento, uma fórmula como hipótese. Como resultado dessas introduções nas inferências, podemos determinar se uma dada fórmula depende ou não de uma certa hipótese. Quando a fórmula inferida se torna independente de alguma das hipóteses, dizemos então que esta descarrega a hipótese *em questão. Na linguagem proposicional, por exemplo, existem tres formas de descarregar as hipóteses:*

1- *Dada uma dedução de β a partir de $\{\alpha\} \cup \Gamma$, podemos inferir $\alpha \rightarrow \beta$ a partir de Γ, sendo então a hipótese α descarregada.*

2- *Dada uma dedução de $\perp$ a partir de $\{\alpha\} \cup \Gamma$, podemos inferir $\neg\alpha$ a partir de Γ, sendo então a hipótese α descarregada.*

3- *Dadas três deduções: uma de $\alpha \vee \beta$, outra de γ a partir de $\{\alpha\} \cup \Gamma_1$, e a terceira de γ a partir de $\{\beta\} \cup \Gamma_2$, podemos inferir γ a partir de $\Gamma_1 \cup \Gamma_2 \cup \{\alpha \vee \beta\}$, sendo α e β as hipóteses descarregadas (α da segunda dedução e β da terceira dedução).*

As regras de inferência consistem de uma regras de introdução e outra de eliminação para cada operador lógico além de duas regras para o $\perp$, uma usada para a lógica intuicionista ($\perp_i$) e outra para a lógica clássica ($\perp_c$)

Para facilitar a leitura da dedução, coloca-se as hipóteses descarregadas entre parênteses, entendendo assim que a nova dedução não depende das hipóteses que foram descarregadas.

$$\frac{\begin{array}{cc}\Pi_1 & \Pi_2 \\ \alpha & \beta\end{array}}{\alpha \wedge \beta}\ \wedge I \qquad \frac{\begin{array}{c}\Pi_1 \\ \alpha \wedge \beta\end{array}}{\alpha}\ \wedge E_r \qquad \frac{\begin{array}{c}\Pi_1 \\ \alpha \wedge \beta\end{array}}{\beta}\ \wedge E_L$$

$$\frac{\begin{array}{c}[\alpha]^u\\ \Pi_1\\ \beta\end{array}}{\alpha \to \beta}\ {\to}I,u \qquad \frac{\begin{array}{cc}\Pi_1 & \Pi_2\\ \alpha \to \beta & \alpha\end{array}}{\beta}\ {\to}E$$

$$\frac{\begin{array}{c}\Pi_1\\ \alpha\end{array}}{\alpha \vee \beta}\ {\vee}I_R \qquad \frac{\begin{array}{c}\Pi_1\\ \beta\end{array}}{\alpha \vee \beta}\ {\vee}I_L \qquad \frac{\begin{array}{ccc}\Pi_1 & [\alpha]^u & [\beta]^v\\ \alpha \vee \beta & \Pi_2 & \Pi_2\\ & C & C\end{array}}{C}\ {\vee}E,u,v$$

$$\frac{\begin{array}{c}\Pi_1\\ \bot\end{array}}{\alpha}\ {\bot}_i \quad \alpha \textit{ diferente de } \bot. \qquad \frac{\begin{array}{c}[\neg\alpha]^u\\ \Pi_1\\ \bot\end{array}}{\alpha}\ {\bot}_c \quad \alpha \textit{ não possui a forma } \beta \to \bot.$$

Nas aplicações de E-regras, a premissa contendo a ocorrência de operadores lógicos que serão eliminados é dita *premissa maior*; a outra premissa é dita *premissa menor* da regra de aplicação.

2.1.1 Corte em Dedução Natural

Durante o processo de construção de uma derivação, não parece ser adequado a introdução de algo e a posterior eliminação em seguida. Este é o conceito chave para simplificação de uma derivação: evitar uma eliminação após uma introdução.

Se uma derivação possui uma introdução seguida por uma eliminação, então pode-se, como regra, simplificá-la.

Apresentaremos os resultados, contudo, eles não serão demonstrados. Maiores detalhes podem ser encontrados em van Dalen (vD97).

Definição 2.4 *Uma fórmula γ é um* corte *em uma derivação quando ela é a conclusão de uma regra de introdução e a premissa maior de uma regra de eliminação. γ é dita a* fórmula do corte.

Derivações podem ser simplificadas através de eliminações de cortes, por exemplo:

$$\frac{\begin{array}{c}\alpha\\ \Pi\\ \beta\end{array} \qquad \begin{array}{c}\Pi'\\ \alpha\end{array}}{\beta}\ {\to}E \qquad \text{converte para} \qquad \begin{array}{c}\Pi'\\ \alpha\\ \Pi\\ \beta\end{array}$$

Denotamos por $\Pi >_1 \Pi'$ para representar "Π converte para Π'". $\Pi > \Pi'$ para representar que "existe uma seqüência finita de conversões $\Pi = \Pi_0 >_1 \Pi_1 >_1 \ldots >_1 \Pi_{n-1} = \Pi$"e $\Pi \geq \Pi'$ para representar $\Pi > \Pi'$ ou $\Pi = \Pi'$. (Π reduz a Π').

Definição 2.5 *Se não existe Π_1' tal que $\Pi_1 >_1 \Pi_1'$ (i.e se Π_1 não contém cortes), então chamamos Π_1 uma derivação normal, ou dizemos que Π_1 está na forma normal, e se $\Pi \geq \Pi'$ onde Π' é normal, então dizemos que Π é normalizado a Π'.*

Dizemos que $>$ possui a propriedade de normalização forte se $>$ não existe uma seqüência infinita de reduções. E $>$ possui a propriedade de normalização fraca se qualquer derivação puder ser normalizada.

Temos que:

Teorema 2.6 (Normalização fraca) *Toda derivação em lógica clássica pode ser normalizada.*

Definição 2.7 *Um* ramo *em uma derivação π é uma seqüência $\alpha_1, \ldots, \alpha_n$ de ocorrências de fórmulas tal que:*

i) *α_1 é uma hipótese;*

ii) *se α_i não é uma premissa menor de $\rightarrow E$ nem a fórmula final de π, então α_{i+1} é a fórmula que ocorre imediatamente abaixo de α_i;*

iii) *α_n é premissa menor de uma aplicação de $\rightarrow E$ ou a fórmula final da dedução.*

O teorema a seguir é uma conseqüência imediata do teorema de normalização.

Teorema 2.8 *Se π é uma derivação normal e $\beta = A_1, \ldots, A_n$ um ramo de π, então temos:*

i) *Uma parte de eliminação de β (possivelmente vazia) $A_1, \ldots, A_{i-1}$ na qual toda fórmula é premissa maior de uma regra de eliminação e contém a fórmula imediatamente sucedente como subfórmula;*

ii) *uma parte mínima A_i de β que é conseqüência de uma regra de eliminação, se $i \neq 1$, e premissa de uma regra de introdução ou de $\perp_c$, se $i \neq n$;*

iii) *Uma parte de introdução de β (possivelmente vazia) $A_{i+1}, \ldots, A_n$ na qual toda fórmula é consequência de uma aplicação de uma regra de introdução e contém a fórmula imediatamente precedente como sua subfórmula.*

Teorema 2.9 (Princípio de Subfórmulas) *Toda ocorrência de uma fórmula em uma dedução normal de A a partir de Γ é subfórmula de A ou de alguma fórmula de Σ, com exceção das hipóteses descarregadas por aplicações de $\perp_c$ e das ocorrências de $\perp$, que estão imediatamente abaixo de tais hipóteses.*

2.2 Classes hierárquicas e problemas lógicos

Para facilitar a compreensão e tornar motivante o estudo de sistemas de provas proposicionais, vamos fazer uma revisão de algumas das teorias de P e NP.

Por convenção, P denota a classe de conjuntos de *strings* reconhecida por uma máquina de Turing determinística em tempo limitado por um polinômio no tamanho da entrada. NP é o mesmo para máquinas de Turing não determinísticas.

Se $TAUT$ denotar o conjunto de tautologias sobre qualquer conjunto fixo adequado de conectivos, temos que $P = NP$ se e somente se $TAUT$ está em P (Pap94).

Agora $P = NP$ não somente pode implicar a existência de algoritmos relativamente rápidos para qualquer algoritmo interessante e aparentemente sem solução em NP, como também pode ter uma conseqüência filosófica interessante em matemática. Se $P = NP$, então existe um polinômio p e um algoritmo A com a propriedade de que qualquer proposição S da teoria de conjuntos e qualquer inteiro n, A determina com apenas $p(n)$ passos se S possui uma prova de tamanho n, ou menor.

Para ver que a existência de A origina-se de $P = NP$, observe que o problema solucionado por A está em NP.

De fato, uma máquina de Turing não-determinística pode escrever qualquer string de tamanho n em sua fita e, então, verificar se a string é uma prova da proposição dada. Para qualquer teoria lógica razoável, esta verificação pode ser realizada em tempo limitado por um polinômio em n. Assim, a importância de mostrar que $P \neq NP$.

Outra importante questão decisiva está relacionada com o fato de NP ser fechado ou não sobre complementação, i. e., $\Sigma^* - L$ está em NP se L está em NP. Se NP não é fechado sobre complementação, então claramente $P \neq NP$ (Pap94). Assim o seguinte resultado é importante:

Teorema 2.10 *NP é fechado sob complementação se e somente se $TAUT$ está em NP.*

Notação 2.11 *$\mathcal{L}$ é o conjunto de funções $f : \Sigma_1^* \to \Sigma_2^*, \Sigma_1, \Sigma_2$, quaisquer alfabeto finito, tal que f pode ser computada por um máquina de Turing determinística em tempo limitado polinomialmente no tamanho da entrada.*

Prova.(do teorema 2.10) O complemento do conjunto de tautologias está em NP, uma vez que para verificar que a fórmula não é uma tautologia pode-se atribuir um valor verdade e verificar que este falsifica a fórmula. Inversamente, suponha que o conjunto de tautologias está em NP. Como qualquer conjunto L em NP é reduzido ao complemento das tautologias no sentido que existe uma função f em $\mathcal{L}$ tal que

para todas strings x, $x \in L$ sse $f(x)$ não é uma tautologia (vide (Pap94)). Assim, um procedimento não determinístico para aceitar o complemento de L é: sob entrada x, compute $f(x)$, e aceite x se $f(x)$ é uma tautologia, usando o procedimento não determinístico para tautologias assumido acima. Assim o complemento de L está em NP. ■

Por convenção utiliza-se $CoNP$ para representação do complemento de NP. Desta forma o teorema 2.10 pode ser enunciado da seguinte forma:

Teorema 2.12 *$NP = CoNP$ se e somente se $TAUT \in NP$*

A questão se $TAUT$ está em NP é equivalente a de existir um sistema de prova proposicional em que toda tautologia possui uma prova curta, formalmente:

Definição 2.13 (Sistema de Prova) *Seja $L \subseteq \Sigma^*$, um* sistema de prova *para L é uma função $f : \Sigma_1^* \to L$ para algum alfabeto Σ_1 e f em L tal que f é sobrejetiva.*

Dizemos que o sistema de prova é limitado polinomialmente *sse existe um polinômio $p(n)$ tal que para todo $y \in L$ existe $x \in \Sigma_1^*$ tal que $y = f(x)$ e $|x| \leq p(|y|)$, onde $|z|$ denota o tamanho de uma string z.*

Exemplo 2.14 (Sistema de prova proposicional) *Para que a noção de sistema de prova proposicional (maiores detalhes são fornecidos em A.1) se encaixe com a definição (2.13), primeiro devemos notar que as fórmulas podem ser consideradas como strings sobre um alfabeto finito.*

O único problema é que uma variável proposicional também deve ser considerada como uma string de forma que exista um suprimento ilimitado de variáveis proposicionais.

Então uma prova π no sistema proposicional que é uma seqüência de fórmulas, pode ser naturalmente considerada como uma string sobre um alfabeto finito que inclui a vírgula como um símbolo separador bem como os símbolos necessários para especificação das fórmulas.

A função f que abstratamente específica o sistema será dada por $f(\pi) = A$ se π provar A, e $f(\pi) = A_0$ para alguma tautologia fixa A_0 se π não é uma string que corresponda a uma prova no sistema.

Teorema 2.15 *Um conjunto L está em NP sse $L = \emptyset$ ou L tem um sistema de provas limitado polinomialmente.*

Prova. Se $L \in NP$, então alguma máquina de Turing não determinística M que aceita L em tempo polinomial. Se $L \neq \emptyset$, definimos f tal que se x codifica a computação de M que aceita y, então $f(x) = y$. Se x não codifica uma computação que aceita, então $f(x) = y_0$ para algum $y_0 \in L$ fixo. Então f é claramente um sistema de prova para L limitado polinomialmente.

Para a volta, se f é um sistema de prova para L limitado polinomialmente, então um algoritmo não-determinístico rápido para aceitar L é, sob entrada y, atribua uma prova curta x de y e verifique $f(x) = y$. ■

Podemos então inferir a partir dos teoremas acima que NP é fechado sob complementação se e somente se $TAUT$ tiver um sistema de provas limitado polinomialmente.

Com o intuito de verificar se os sistemas de provas convencionais são limitados polinomialmente, Cook e Rechkow em (CR79) definiram classes de equivalências para sistemas de provas de forma que a resposta fosse a mesma para sistemas equivalentes.

A relação de equivalência associada é a p-simulação.

Notação 2.16 *$\mathcal{L}$ é o conjunto de funções $f : \Sigma_1^* \rightarrow \Sigma_2^*, \Sigma_1, \Sigma_2$ qualquer alfabeto finito, tal que f possa ser verificada por uma máquina de Turing determinística em tempo limitado por um polinômio no tamanho da entrada.*

Existem duas noções mais comuns de comprimento de prova. A primeira é o número de linhas que aparecem em uma prova, usualmente chamada *número de linhas* ou *número de inferências* de uma prova.

A segunda é a noção de número de símbolos aparecendo na prova e esta, como já foi mencionado no começo desta seção, será a medida que utilizaremos no decorrer do texto.

Definição 2.17 *Se $f_1 : \Sigma_1^* \rightarrow L$ e $f_2 : \Sigma_2^* \rightarrow L$ são sistemas de provas para L, então f_2* p-simula *f_1 se existir uma função $g : \Sigma_1^* \rightarrow \Sigma_2^*$ tal que g está em $\mathcal{L}$, e $f_2(g(x)) = f_1(x)$ para todo x.*

Segue da definição acima e da definição (2.13) que:

Teorema 2.18 *Se um sistema de prova para L, f_2, p-simula um outro sistema de provas para L, f_1, limitado polinomialmente então f_2 também é limitado polinomialmente.*

2.3 Sistemas de Frege

Nesta seção definiremos sistemas de Frege, sistemas de Frege com substituição e sistemas de Frege estendidos. Alguns resultados envolvendo sistemas de Frege e p-simulação também serão apresentaados. Maiores informações são apresentadas no apêndice A.

Definição 2.19 *Um* sistema de Frege*, $\mathcal{F}$, é definido por um conjunto completo de conectivos proposicionais, tendo um conjunto finito de regras de inferência esquematicamente definido. O sistema admite* modus ponens *como uma regra derivada e precisa ser completo e consistente.*

Um sistema de Frege com Substituição*, $\mathcal{SF}$, é definido por adicionar a $\mathcal{F}$ a seguinte regra de substituição*

$$\frac{A(p)}{A(B)}$$

onde simultâneas substituições da fórmula B são permitidas para a variável p.

Definição 2.20 (Sistema de Frege estendido) ,

Um sistema de Frege estendido*, e-$\mathcal{F}$, é obtido por adicionar a $\mathcal{F}$ a seguinte regra:*

$$p \leftrightarrow A$$

onde A é qualquer fórmula e p é uma variável que não ocorre nem A nem em qualquer regra de extensão usada na prova e nem na fórmula final da prova.

A idéia da regra de extensão é permitir que a variável p aja como uma abreviação para a fórmula A.

Teorema 2.21 *Um sistema de Frege estendido é limitado polinomialmente se e somente se todos os sistemas de Frege estendidos sob todos os conjuntos de conectivos forem limitados polinomialmente. Além disso, um sistema de Frege estendido é polinomialmente limitado se e somente se existir um limite polinomial sob o número de linhas nas provas em $e\mathcal{F}$. Desta forma, se $P \neq NP$, então não existe um limite polinomial sob o número de linhas em provas em sistemas de Frege estendidos, sistemas de Frege ou sistema de Dedução Natural.*

2.4 Resolução

O sistema de Resolução trabalha com variáveis proposicionais $p_1, p_2 \ldots$ e suas negações $\neg p_1, \neg p_2, \ldots$ e não tem nenhum outro conectivo lógico, ou seja, $\kappa = \emptyset$.

O objeto básico em Resolução é uma cláusula, um conjunto finito de literais (variáveis proposicionais ou variáveis proposicionais negadas). Ou seja, uma linha em resolução é um cláusula.

Uma *cláusula* pode ser vista como uma disjunção de *literais*. Desta forma, uma atribuição de verdade satisfaz uma cláusula C se e somente se satisfizer pelo menos um literal de C. A *Regra de resolução* nos permite derivar novas cláusulas $C_1 \cup C_2$ de duas cláusulas $C_1 \cup \{p\}$ e $C_2 \cup \{\neg p\}$

$$\frac{C_1 \cup \{p\} \qquad C_2 \cup \{\neg p\}}{C_1 \cup C_2}$$

Não existem restrições sob ocorrências de literais $p, \neg p$ em C_1 ou C_2. Uma propriedade óbvia de resolução é que, se uma atribuição de verdade satisfaz as duas cláusulas superiores da regra, então ela também satisfaz a cláusula inferior.

Uma refutação em resolução de um conjunto de cláusulas $\mathcal{C} = \{C_1, \ldots, C_n\}$ é uma seqüência de cláusula $D_1, \ldots, D_t$ tal que cada D_i é ou um elemento de $\mathcal{C}$ ou é derivado por regra de resolução de alguns $D_u, D_v, u, v < i$ anteriores, e tal que a última cláusula é a cláusula vazia.

Teorema 2.22 *Um conjunto de cláusulas é insatisfatível: isto é, não existe nenhuma atribuição de verdade que simultaneamente satisfaça todas as cláusulas no conjunto, se e somente se existir uma refutação em resolução do conjunto.*

2.5 Princípio das Casas dos Pombos-PHP

Tautologias expressando versões do princípio das casas dos pombos têm sido importantes casos usados para teste com o intuito de obter limites no comprimento de provas proposicionais e de comparar o poder de prova dos vários sistemas proposicionais.

O primeiro artigo de Cook e Rechkow (CR79) mostrou que o princípio das casas dos pombos possui prova de tamanho polinomial em sistemas de Frege estendido. Buss em (Bus87) mostrou que o princípio também possui uma prova polinomial em sistema de Frege.

Já Haken em (Hak85) mostrou que uma refutação em resolução do princípio das casas dos pombos requer um tamanho exponencial.

Apresentaremos, nesta seção, o princípio das casas dos pombos proposicional e um esquema das provas em $\mathcal{F}$ e e-$\mathcal{F}$, uma vez que o utilizaremos nas próximas seções para aplicar nosso resultado.

2.5.1
PHP_n

Para cada número natural $n \geq 1$, seja PHP_n a fórmula proposicional que expressa que "se $n+1$ pombos descansam em n casas então alguma casa contém mais que um pombo". Formalmente, se $[n]$ denota o conjunto $\{0, 1, \ldots, n-1\}$; se $f : [n+1] \to [n]$ então existem $0 \leq i < j \leq n$ tal que $f(i) = f(j)$. Para expressar proposicionalmente utilizaremos a variável proposicional $p_{i,j}$ significando $f(i) = j$ e definiremos PHP_n pela fórmula.

$$\bigwedge_{i=0}^{n} \bigvee_{j=0}^{n} p_{i,j} \to \bigvee_{0 \leq i < m \leq n} \bigvee_{j=0}^{n} (p_{i,j} \wedge p_{m,j})$$

O lado esquerdo da fórmula expressa que a função é total e o lado direito que a função não é injetiva. Note que o tamanho de PHP_n é proporcional a n^3.

Uma prova informal do princípio pode ser dada por uma indução sobre n. É claro que o princípio é verdadeiro para $n = 2$.

Em geral, se $f : [n+1] \to [n]$, então seja $f' : [n] \to [n-1]$ definida por

$$f'(i) = \begin{cases} f(i), \text{ se } f(i \neq n-1) \\ f(n), \text{ caso contrário} \end{cases}$$

Se f é injetiva, é fácil verificar que f' também o é, contradizendo a hipótese de indução.

Para formalizar esta prova em sistema de Frege estendido e-$\mathcal{F}$, usamos novas variáveis proposicionais $q^k_{i,j}$ para representar a asserção $f_k(i) = j$. Isto é, definiremos $q^k_{i,j}$ como segue:

$$\begin{cases} q^k_{i,j} \leftrightarrow p_{i,j}, & 0 \leq i \leq n, 0 \leq j < n, \\ q^n_{i,j} \leftrightarrow q^{k+1}_{i,j} \vee (q^{k+1}_{i,k} \wedge q^{k+1}_{k+1,j}), & 0 \leq i \leq k, 0 \leq j < k, 1 \leq k < n. \end{cases}$$

Seja A_k a fórmula proposicional

$$\bigwedge_{i=0}^{k} \bigvee_{j=0}^{k} q^k_{i,j} \to \bigvee_{0 \leq i < m \leq k} \bigvee_{j=0}^{k} (q^k_{i,j} \wedge q^k_{m,j})$$

Fica claro que existe uma prova de $\neg PHP_n \to \neg A_n$ e $\neg A_{k+1} \to \neg A_k$ para todo $1 \leq k < n$ em que cada prova possui tamanho $O(n^6)$. Este tamanho estimado é obtido através da observação de que existe uma prova em e-$\mathcal{F}$ de $\neg A_{k+1} \to A_k$ com $O(n^3)$ linhas e cada fórmula nesta prova tem tamanho $O(n^3)$.

Uma vez que existe uma prova em e-$\mathcal{F}$ de $A_1 = q^1_{0,0} \wedge q^1_{1,0} \to q^1_{0,0} \wedge q^1_{1,0}$. Usando *Modus Ponens* n vezes para combinar as provas em e-$\mathcal{F}$ de $\neg PHP_n \to$

$\neg A_n$ e $\neg A_{k+1} \rightarrow \neg A_k$ obtemos uma prova em e-$\mathcal{F}$ de PHP_n de tamanho $O(n^7)$.

Uma forma simples de converter esta prova em e-$\mathcal{F}$ para uma prova em $\mathcal{F}$ é substituir cada variável proposicional introduzida pela regra de extensão pela fórmula que ela abrevia. A nova prova terá o mesmo número de linhas. Entretanto, o tamanho das fórmulas $q^n_{0,0}$ e $q^1_{1,0}$ apresentará um tamanho de ordem 3^n. Assim, o tamanho da prova em $\mathcal{F}$ será $O(n^4.3^n)$.

Acreditava-se que o princípio das casas dos pombos seria um exemplo que mostra a separação exponencial entre e-$\mathcal{F}$ e $\mathcal{F}$. Entretanto, Buss em (Bus87) provou que PHP_n possui uma prova polinomial também em $\mathcal{F}$.

Para mostrar este resultado Buss apresenta uma prova polinomial em e-$\mathcal{F}$ de *PHP* cujas regras de extensão, ao serem substituídas, geram uma prova em $\mathcal{F}$ que continua sendo polinomial.

A estratégia, para a escolha das regras de extensão, envolve vetor adição e vetor contagem para circuitos de profundidade logarítmica que aparecem em Ofman (Ofmry) e Wallace(Wal64).

2.6 PHP em Resolução

Apresentaremos um resultado devido Haken, Buss e Turán (Kra95) no qual o tamanho mínimo de uma refutação em resolução de $\neg PHP$ é exponencial.

Se considerarmos as variáveis proposicionais $p_{ij}, i = 0, \ldots, n$ e$j = 0, \ldots, n-1$ então $\neg PHP_n$ pode ser considerado como o conjunto de cláusulas

$$\{\neg p_{ik}, \neg p_{jk}\}, \text{ para todo } i \neq j \text{ e } k$$
$$\{p_{i0}, \ldots, p_{i(n-1)}\}, \text{ para todo } i.$$

Teorema 2.23 *Em qualquer refutação em resolução do conjunto $\neg PHP_n$ existem pelo menos $2^{\Omega(\frac{n^2}{m})}$ cláusulas diferentes.*

Como o princípio das casas dos pombos possui prova polinomial tanto em $\mathcal{F}$ quanto em e-$\mathcal{F}$, concluímos que Resolução não é nem um sistema de Frege e nem um sistema de Frege estendido.

Uma vez que o sistema de resolução pode ser visto como um cálculo de sequentes com cortes de tamanho 1, temos que o teorema acima nos reforça a idéia de que a introdução de cortes de tamanho maior que 1 seria uma alternativa para reduzir o tamanho de provas proposicionais.