Uma Introdução à Teoria da Prova

Ana Teresa Martins[1] **, Anjolina G. de Oliveira**[2] [*]**, Ruy J. G. B. de Queiroz**[2]

[1]Laboratório de Inteligência Artificial – Departamento de Computação
Universidade Federal do Ceará – C.P. 12.166, Fortaleza - Ce

[2]Centro de Informática – Universidade Federal de Pernambuco
C.P. 7851, Recife - Pe

ana@lia.ufc.br, ago@cin.ufpe.br, ruy@cin.ufpe.br

Abstract. *The study of structural properties of formal proofs in Logic is at the heart of research in Proof Theory. The characterization of normal proofs, the existence, unicity and consistency theorems about normal proofs, the estimation of limits for proof length of a certain formula are, among others, results achieved in this area. The development of Automatic Theorem Provers and Logic/Functional Programming has benefitted from these results. An introduction to this topic is here presented through systems of natural deduction and sequent calculus for classical and intuicionistic logic, and via the advances of proof theory in linear logic as well as through the connections between proofs and computational processes. Finally, a brief introduction to Type Theory is presented.*

Resumo. *O estudo das propriedades estruturais de provas formais em Lógica constitui o cerne da pesquisa relacionada à Teoria da Prova. A caracterização da noção de prova normal, os teoremas de existência, unicidade e consistência de tais provas, o estabelecimento de limites para o tamanho da prova de uma dada fórmula são, dentre outros, resultados alcançados nesta área. O desenvolvimento de Provadores Automáticos de Teoremas e da Programação em Lógica e Funcional tem sido impulsionado por tais resultados. Uma introdução ao tema é aqui apresentada através dos sistemas de dedução natural e cálculo de seqüentes para as lógicas clássica e intuicionística, dos avanços da teoria da prova no escopo da lógica linear, assim como da relação entre provas e processos computacionais. Por fim, apresenta-se uma breve introdução à Teoria dos Tipos.*

1. Introdução

Entre os meados do século XIX e início do século XX, com o desenvolvimento dos métodos algébricos e da teoria dos conjuntos assim como com o descobrimento de diversos paradoxos envolvendo conceitos conjuntistas, a fundamentação da matemática passou a ser o foco central de três escolas denominadas: Logicismo, Intuicionismo e

[*]O autor realizou parte desse trabalho enquanto foi integrante do Centro de Processamento de Dados e Laboratório de Sistemas Distribuídos (DCC) da Universidade Federal da Bahia.

Formalismo. Esta última, liderada por David Hilbert, pretendia provar a consistência de axiomatizações da matemática através de métodos considerados recursivos, finitistas, computacionais [Cos92, Kle52].

Apesar do resultado negativo do Teorema da Incompletude de Gödel, variações do programa de Hilbert continuaram a ser perseguidos. Por um lado, buscava-se a prova da autoconsistência da matemática, ou seja, buscava-se a prova da consistência da matemática por métodos lógico-matemáticos. Neste contexto, as provas formais passaram a ser objeto de estudo do que se passou a chamar de Teoria da Prova Estrutural, ou simplesmente Teoria da Prova [TH96]. Por outro lado, modelos formais foram desenvolvidos visando capturar a noção de procedimento recursivo, finitista, computacional. À conjectura de que tais modelos formais capturam o conceito pretendido de computação e à prova da equivalência de tais modelos chama-se Tese de Church. Tal resultado foi fundamental para o desenvolvimento da Teoria da Computação pois estabeleceu os limites do que se pode computar e do que não dá para ser computável [DSW94].

O estudo das propriedades estruturais das provas formais envolve o estudo do tamanho da prova de uma dada fórmula, do critério de unicidade, existência e equivalência de provas, entre outros. Os resultados obtidos no âmbito da Teoria da Prova têm impulsionado o desenvolvimento de Provadores Automáticos de Teoremas e da Programação em Lógica e Funcional [GLT89], como também têm possibilitado a caracterização de classe de problemas sob a perspectiva da Complexidade Computacional [Gir87b].

São apresentados, a seguir, os sistemas de dedução natural e cálculo de seqüentes para as lógicas clássica e intuicionística assim como os principais teoremas acerca das propriedades estruturais das provas formais. São também introduzidos os avanços da teoria da prova no escopo da lógica linear assim como é estabelecida uma relação entre provas e processos computacionais.

2. Dedução Natural

Em [Gen35], Gentzen propõe uma teoria da dedução baseada nas regras de manuseio de suposições em lugar da usual formulação axiomática na qual praticamente não há lugar para o raciocínio hipotético. O seu propósito era construir um sistema formal que se aproximasse ao máximo do modo de raciocínio humano. Como resultado desse propósito, Gentzen apresenta um cálculo de dedução para a lógica intuicionística, o cálculo **NJ**, e um cálculo para a lógica clássica, o cálculo **NK**. Esses sistemas ficaram conhecidos como sistemas de dedução natural (DN). No sistema de dedução natural as regras de inferência são projetadas num padrão de regras de introdução e eliminação, que seguem o *princípio da inversão*: a regra de eliminação é o inverso da regra de introdução. A partir desse princípio, Prawitz [Pra65, Pra71a] analisa as principais propriedades dos sistemas de dedução natural e apresenta uma das suas principais contribuições para a dedução natural: a normalização de provas.

2.1. Os fragmentos mínimo e intuicionístico

As regras de inferências para o fragmento proposicional são mostradas na tabela 1. As proposições acima da linha são chamadas de premissas e a que está abaixo conclusão. Aqui é adotada a linguagem proposicional usual incluindo as constantes lógicas: $\wedge$ (conjunção); $\vee$ (disjunção); $\rightarrow$ (implicação); e a constante Λ para o *absurdo* ou *falso*. As

regras $\rightarrow$-I e $\vee$-E são chamadas de " regras impróprias" por Prawitz [Pra71a] porque elas *cancelam* (descartam) ou *fecham* suposições, que são escritas entre colchetes. Nas regras que possuem a negação, é possível observar que $\neg A$ pode ser vista como uma abreviação de $A \rightarrow \Lambda$. Nesse caso, as regras $\neg$-I e $\neg$-E se tornam casos especias das regras de introdução e eliminação da implicação ($\rightarrow$), respectivamente.

O sistema mostrado na tabela 1 é chamado de *mínimo* ($\mathbf{M}$) [Pra71a]. O cálculo NJ (sistema $\mathbf{I}$) é obtido a partir do sistema $\mathbf{M}$ adicionando-se a seguinte regra (Λ_I): $\dfrac{\Lambda}{A}$. Essa regra é conhecida como a *regra do absurdo intuicionístico* ou *regra de negação intuicionística* ou *ex falsum quodlibet*.

$\dfrac{A \quad B}{A \wedge B}\ \wedge$-*I*	$\dfrac{A \wedge B}{A}\ \wedge$-*E1* $\quad \dfrac{A \wedge B}{B}\ \wedge$-*E2*
$\dfrac{A}{A \vee B}\ \vee$-*I1* $\quad \dfrac{B}{A \vee B}\ \vee$-*I2*	$\dfrac{\begin{array}{ccc} & [A] & [B] \\ A \vee B & \vdots & \vdots \\ & C & C \end{array}}{C}\ \vee$-*E*
$\dfrac{\begin{array}{c}[A] \\ \vdots \\ B\end{array}}{A \rightarrow B}\ \rightarrow$-*I*	$\dfrac{A \quad A \rightarrow B}{B}\ \rightarrow$-*E*
$\dfrac{\begin{array}{c}[A] \\ \vdots \\ \Lambda\end{array}}{\neg A}\ \neg$-*I*	$\dfrac{A \quad \neg A}{\Lambda}\ \neg$-*E*

Tabela 1: Regras de inferência para o fragmento proposicional

Definição 1 (premissa maior e menor) *Nas regras de eliminação, a premissa que contem a constante lógica é chamada de premissa maior, a(s) outra(s) premissa(s) (se existir(em)) é(são) chamada(s) de premissa menor.*

As regras de inferência são os passos atômicos das derivações. Na seção 2.3. os exemplos dados ilustram o seu uso na construção de derivações.

Introdução e eliminação do $\wedge$. Essas regras são bastante intuitivas. Se temos A e B então podemos concluir $A \wedge B$ ($\wedge$-*I*). Se temos $A \wedge B$ podemos concluir ou A ($\wedge$-E_1) ou B ($\wedge$-E_2).

Introdução e eliminação do $\vee$. As regras de introdução do $\vee$ são imediatas. Se temos A então podemos deduzir $A \vee$ "qualquer fórmula", por exemplo B, nesse caso usamos a regra $\vee$-I_1. De forma similar, se temos B podemos deduzir "qualquer fórmula" (por ex. A) $\vee\ B$ pela regra $\vee$-I_2. Entretanto, a regra de eliminação do $\vee$ não é tão evidente quanto às regras de introdução. Para entender melhor o seu significado, vamos supor por

exemplo que a proposição A seja "como muito macarrão" e a proposição B seja "como muito bolo" . Não importa qual das duas ou se as duas proposições são verdadeiras, o efeito vai ser o mesmo: "irei engordar" (proposição C). Portanto, na eliminação do $\vee$, se temos $A \vee B$, e supondo que temos A concluimos C, e sunpondo B deduzimos C, então podemos concluir C, e cancelar as suposições A e B (não precisamos mais dessas suposições).

Introdução e eliminação do $\rightarrow$. A regra da introdução do $\rightarrow$ diz que se deduzimos B a partir da suposição A então podemos concluir $A \rightarrow B$ e não precisamos mais da suposição A, ou seja a suposição foi cancelada. É importante observar que essa regra juntamente com a regra de eliminção do $\vee$ são diferentes das outras por ter uma característica *global*, ou seja, a premissa da regra não é simplesmente uma fórmula, mas sim uma derivação. No caso da regra $\vee$-*E*, as duas premissas menores são derivações de C. A regra de eliminação do $\rightarrow$ é fácil de entender. Se temos A e a partir de A deduzimos B ($A \rightarrow B$) então podemos concluir que temos também B. Ainda sobre a regra de introdução do $\rightarrow$, a derivação $\dfrac{B}{A \rightarrow B}$ é correta. Nesse caso não há cancelamento de suposições. É importante observar também que as regras que cancelam suposições podem fechar mais de uma ocorrência da mesma fórmula através de uma única aplicação da regra.

Introdução e eliminação da negação. Se supomos A e chegamos à uma contradição (absurdo: Λ), então não é o caso de termos A, logo concluímos $\neg A$. A eliminação da negação é bastante evidente, se temos ao mesmo tempo A e $\neg A$ então temos uma contradição (Λ).

Regra do absurdo intuicionístico. Se temos o absurdo (Λ) então podemos deduzir "qualquer coisa".

2.2. O cálculo clássico

Para obter o cálculo clássico (NK) a partir do cálculo NJ, Gentzen [Gen35] propôs duas formas:

1. através da inclusão da "lei do terceiro excluído" ($A \vee \neg A$, onde A pode ser qualquer fórmula arbitrária) como um esquema de axioma;
2. ou através da adição de uma nova regra: $\dfrac{\neg\neg A}{A}$

A última alternativa representa uma nova eliminação da negação ao mesmo tempo em que é bastante diferente do padrão das outras regras do sistema. Gentzen preferia a primeira opção, para ele era uma alternativa "mais natural" ([Gen35], p. 81).

Em [Pra65], Prawitz define um sistema para DN clássica (chamado $\mathbf{C}$) adicionando a *regra do absurdo clássico* (Λ_C) (também chamada de *princípio da prova indireta*): $\dfrac{\begin{array}{c}[\neg A] \\ \vdots \\ \Lambda\end{array}}{A}$ ao sistema mínimo ($\mathbf{M}$). Essa regra é a formalização de provas feitas a partir de

uma contradição, ou seja as conhecidas provas por refutação: Se negamos A e deduzimos o absurdo então provamos A.

Para estudar a forma normal de derivações, Prawitz considera um sistema menor para a lógica clássica, no qual as constantes $\vee$ e $\exists$ são excluídas e as aplicações da regra Λ_C são restringidas para o caso onde A é uma fórmula atômica e diferente de Λ.

Observa-se que Prawitz seguiu a segunda alternativa dada por Gentzen para apresentar o sistema de DN clássica. Desde que $\neg\neg A$ é equivalente a A na lógica clássica, tem-se que se a partir da suposição $\neg A$ obtem-se Λ (*o falso*), então chega-se a prova de A. Isso acontece devido a regra de introdução da implicação e das correspondências entre $\neg A$ e $A \rightarrow \Lambda$, e $\neg\neg A$ e A.

Existem alguns trabalhos que exploram a primeira alternativa proposta por Gentzen ao introduzir regras para a lei do terceiro excluído. Os trabalhos de J. Seldin [Sel89] e J. von Plato [Pla98] seguem essa linha.

2.3. Derivações

As provas nos sistema de DN são representadas por derivações escritas em forma de árvore, onde: (i) as fórmulas do topo da árvore são chamadas de *suposições*; (ii) as outras fórmulas são obtidas das fórmulas que estão imediatamente acima através de uma das regras de inferência.

A seguinte notação é usada por Prawitz [Pra71a]:

- Π denota derivações enquanto que Σ sequências finitas de derivações;
- $\Gamma \vdash A$ indica que A é *derivável* a partir de Γ, *i.e.* existe uma árvore de dedução contruída a partir das regras do sistema, na qual a fórmula final A depende somente das fórmulas de Γ. Γ é o conjunto de suposições. Se Γ é vazio, diz-se apenas que A é derivável;
- É usada uma operação de concatenação ($\Sigma/\Gamma/\Pi$), onde Γ é um conjunto de fórmulas do topo de Π. As fórmulas de Γ estão imediatamente abaixo das fórmulas finais de Σ. Quando o conjunto de fórmulas Γ tiver a forma A, usa-se $[A]$ para denotar o conjunto: $\dfrac{\Sigma}{\dfrac{[A]}{\Pi}}$

2.3.1. Exemplos

A partir dos exemplos aqui apresentados será introduzida uma técnica de construção de derivações em DN. Algumas regras "quebram" as fórmulas (as regras de eliminação) enquanto outras "juntam" fórmulas (as regras de introdução). Dessa forma ao construir derivações quebramos as fórmulas para depois juntá-las novamente, como se fosse um jogo. Na seção seguinte, veremos através do *teorema da normalização* que sempre há uma "estratégia" boa para o jogo, de forma a não haver redundâncias.

Exemplo 1 *Seja o seguinte teorema: $A \wedge B \rightarrow A$. Para provar $A \wedge B \rightarrow A$ é preciso supor $A \wedge B$ e tentar derivar A. Ao construir derivações em DN, temos sempre que olhar para as suposições que dispomos e para o resultado que queremos atingir. Nessa etapa,*

ao aplicarmos uma regra de eliminação do $\wedge$ à proposição $A \wedge B$ podemos obter A ou B. Mas, o nosso objetivo é A, portanto a regra a ser aplicada é $\wedge$-E$_1$. As suposições são marcadas por um índice que é utilizado para indicar o seu cancelamento ao lado das regras que as fecham. A derivação de $A \wedge B \rightarrow A$ é mostrada a seguir:

$$\dfrac{\dfrac{[A \wedge B]^1}{A}\wedge\text{-}E_1}{A \wedge B \rightarrow A} \rightarrow\text{-}I_{(1)}$$

Exemplo 2 *Seja o seguinte teorema: $A \rightarrow (\neg A \rightarrow B)$. Começamos a derivação supondo A, pretendemos alcançar $\neg A \rightarrow B$. Aqui temos uma segunda suposição: $\neg A$ e o nosso objetivo é B, que é uma fórmula que não pode ser "quebrada". Agora é a fase de juntar o que temos para construir a derivação desejada. Se temos as suposições A e $\neg A$ concluímos o falso (Λ) pela regra $\neg$-E. Se temos o absurdo, pela regra do absurdo intuicionístico concluímos B (nosso objetivo). Agora cancelamos as suposições de forma a obter o resultado desejado. Primeiro cancelamos $\neg A$ e depois A, ambas são fechadas pela regra $\rightarrow$ -I. A derivação é mostrada a seguir:*

$$\dfrac{\dfrac{\dfrac{[A]^1 \quad [\neg A]^2}{\dfrac{\Lambda}{B}\Lambda_I}\neg\text{-}E}{\neg A \rightarrow B} \rightarrow\text{-}I_{(2)}}{A \rightarrow (\neg A \rightarrow B)} \rightarrow\text{-}I_{(1)}$$

Exemplo 3 *As seguintes derivações ilustram o uso da regra Λ_C e da regra $\vee$-E.*

$$\dfrac{\dfrac{\dfrac{[\neg\neg A]^1 \quad [\neg A]^2}{\dfrac{\Lambda}{A}\Lambda_{C(2)}}\neg\text{-}E}{\neg\neg A \rightarrow A} \rightarrow\text{-}I_{(1)}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{[P \vee Q]^3 \qquad \dfrac{\dfrac{[P \rightarrow R]^1 \quad [P]^4}{R} \rightarrow\text{-}E \qquad \dfrac{[Q]^5 \quad [Q \rightarrow R]^2}{R} \rightarrow\text{-}E}{R}\vee\text{-}E_{(4,5)}}{P \vee Q \rightarrow R} \rightarrow\text{-}I_{(3)}}{Q \rightarrow R \rightarrow ((P \vee Q) \rightarrow R)} \rightarrow\text{-}I_{(2)}}{(P \rightarrow R) \rightarrow ((Q \rightarrow R) \rightarrow ((P \vee Q) \rightarrow R))} \rightarrow\text{-}I_{(1)}$$

2.4. Normalização por Prawitz

2.4.1. O princípio da inversão e as reduções

A partir de uma análise da estrutura das derivações em DN, Prawitz define o *princípio da inversão* que reflete as simetrias existentes entre as regras de introdução e a correspondente regra de eliminação nos sistemas de dedução natural. Baseado nesse princípio, Prawitz define um conjunto de *regras de redução* que formam o *procedimento de normalização*, utilizado para se obter a *forma normal* das derivações.

Em [Pra65, Pra71a], Prawitz observa que as regras de eliminação, em um certo sentido, são o inverso da correspondente regra de introdução. Se a premissa maior da regra de eliminação foi deduzida a partir da regra correspondente de introdução, então nada de novo foi acrescentado à prova. O *princípio da inversão* define essa simetria:

Definição 2 (princípio da inversão) *Seja E uma aplicação de uma regra de eliminação que possui B como conseqüência. Se as deduções utilizadas para derivar a premissa maior de E quando combinadas com as deduções da premissa menor de E (se existir), já contiverem a dedução de B, então, a dedução de B pode ser obtida diretamente a partir das deduções das premissas de E sem a aplicação de E.*

Como observado por Prawitz, essas idéias já estavam contidas nos trabalhos de Gentzen, quando ele diz que uma regra de introdução dá a definição da constante lógica e a regra de eliminação é somente uma conseqüência da correspondente regra de introdução. Entretanto, foi Prawitz quem formulou a definição precisa do princípio da inversão, através da apresentação de um conjunto de *regras de redução*.

Os seguintes exemplos ilustram o princípio da inversão:

Exemplo 4 *Seja a seguinte derivação:*

$$\dfrac{\dfrac{\dfrac{\Sigma_1}{A} \quad \dfrac{\Sigma_2}{B}}{\boxed{A \wedge B}} \wedge\text{-}I}{A} \wedge\text{-}E$$

A conclusão dessa derivação (A), obtida pela eliminação do $\wedge$, já está contida na prova da premissa $A \wedge B$, que foi inferida através de uma introdução do $\wedge$.

A fórmula $A \wedge B$ é conseqüência de uma regra de introdução ao mesmo tempo em que é a premissa maior da correspondente regra de eliminação. Isso representa um "desvio" desnecessário que pode ser removido, de forma que a derivação possa ser reduzida para a seguinte:

$$\dfrac{\Sigma_1}{A}$$

Exemplo 5 *Seja a seguinte derivação:*

$$\dfrac{A \qquad \dfrac{\dfrac{[A]}{B}}{\boxed{A \to B}} \to\text{-}I}{B} \to\text{-}E$$

A conclusão (B), obtida pela eliminação do $\to$, já está contida na prova da premissa maior ($A \to B$), que foi inferida através de uma introdução. Em outras palavras, nada de novo é obtido quando se tem uma eliminação logo após uma introdução (na premissa maior da eliminação) do mesmo conectivo.

A partir do *princípio da inversão*, Prawitz observa que as seqüências de inferências, como as mostradas no exemplo, que não introduzem nada de novo na prova, podem

ser dispensadas. Consequentemente, as provas podem ser transformadas para que redundâncias como essas sejam eliminadas. Antes de definir mais precisamente com são essas transformações, Prawitz fornece alguns conceitos importantes.

Definição 3 (Fórmula máxima) *Em uma derivação, a ocorrência de uma fórmula que é ao mesmo tempo a conclusão de uma introdução e premissa maior de uma eliminação é chamada de fórmula máxima.*

Nos exemplos aqui mostrados, as fórmulas destacadas são fórmulas máximas. O princípio da inversão diz que elas se constituem num "desvio" desnecessário, portanto podem ser removidas. Elas são removidas através de um conjunto de transformações entre provas chamado de *reduções próprias*.

Para o fragmento proposicional, são três reduções próprias, uma para cada conectivo. Elas são mostradas na tabela 2.

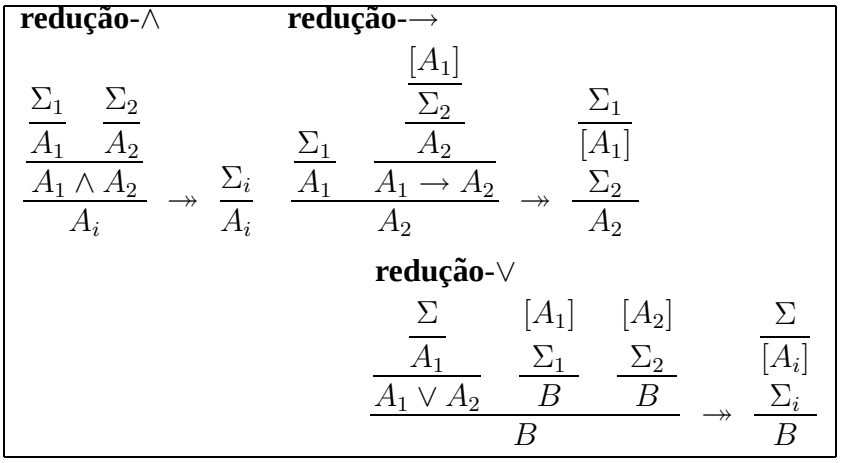

Tabela 2: Reduções próprias

Sobre as reduções próprias, tem-se as seguintes observações:

- **redução**-$\vee$: $[A_i]$ denota o conjunto de suposições em Σ_i, que são fechadas pela regra $\vee$-*E* (i pode ser 1 ou 2).
- **redução**-$\rightarrow$: $[A_1]$ denota o conjunto de suposições em Σ_2, que são fechadas pela regra $\rightarrow$ -*E*.

Um outro tipo de "desvio" pode ocorrer em derivações que possuem aplicações da regra $\vee$-*E* (para a lógica de primeira ordem a regra $\exists$-*E* também é incluída) porque na aplicação das regras de eliminação do $\vee$, ocorrências de fórmulas da mesma fórmula aparecem umas após as outras, formando uma sequência de ocorrências de fórmulas iguais.

A noção de *segmento máximo* é introduzida para caracterizar esse tipo de desvio e as *reduções permutativas* são utilizadas para removê-los.

Definição 4 (Segmento máximo) *Uma sequência de ocorrências de fórmulas $A_1, A_2, ..., A_n$, em uma derivação, é chamada de segmento máximo quando as ocorrências de fórmulas da sequência são ocorrências da mesma fórmula, A_{i+1} ocorre logo após de A_i, A_1 é a conclusão de uma introdução e A_n é a premissa maior de uma eliminação (A_i para $i < n$ é uma premissa menor de $\vee$-E ou $\exists$-E).*

O seguinte exemplo ilustra a ocorrência de um segmento máximo em uma derivação:

Exemplo 6 *Seja a prova:*

$$\dfrac{(A \wedge B) \vee (A \wedge C) \qquad \dfrac{\dfrac{\dfrac{[A \wedge B]}{A}\wedge\text{-}E \qquad D}{\mathbf{A \wedge D}}\wedge\text{-}I \qquad \dfrac{\dfrac{[A \wedge C]}{A}\wedge\text{-}E \qquad D}{\mathbf{A \wedge D}}\wedge\text{-}I}{\mathbf{A \wedge D}}\vee\text{-}E}{A}\wedge\text{-}E$$

As fórmulas em negrito formam um segmento máximo.

Para o fragmento proposicional só existe uma redução permutativa, que se aplica ao conectivo $\vee$, conforme mostrada na tabela 3. A regra indicada por $\mathbf{r}$ é uma regra de eliminação e C é a premissa maior da regra.

redução-permutativa-$\vee$

$$\dfrac{\dfrac{\Sigma_1}{A \vee B} \quad \dfrac{\Sigma_2}{C} \quad \dfrac{\Sigma_3}{C}}{\dfrac{C \quad \Sigma_4}{D}\mathbf{r}} \;\twoheadrightarrow\; \dfrac{\dfrac{\Sigma_1}{A \vee B} \quad \dfrac{\dfrac{\Sigma_2}{C} \quad \Sigma_4}{D}\mathbf{r} \quad \dfrac{\dfrac{\Sigma_3}{C} \quad \Sigma_4}{D}\mathbf{r}}{D}$$

Tabela 3: Redução permutativa

Ao remover segmentos máximos através de reduções permutativas é possível que apareçam fórmulas máximas, conforme ilustrado no exemplo a seguir.

Exemplo 7 *Ao aplicar a redução permutativa do $\vee$ à derivação do exemplo 6, a seguinte derivação é obtida:*

$$\dfrac{\dfrac{\Sigma_1}{(A \wedge B) \vee (A \wedge C)} \qquad \dfrac{\dfrac{[A \wedge B]}{A} \quad \Sigma_2}{\mathbf{A \wedge D}}\quad \dfrac{\dfrac{[A \wedge C]}{A} \quad \Sigma_2}{\mathbf{A \wedge D}}}{A}$$

Observamos que "aparece" uma fórmula máxima, destacada em negrito. Através de uma redução própria, a fórmula màxima $A \wedge D$ pode ser removida:

$$\dfrac{\dfrac{\Sigma_1}{(A \wedge B) \vee (A \wedge C)} \qquad \dfrac{[A \wedge B]}{A} \quad \dfrac{[A \wedge C]}{A}}{A}$$

Definição 5 (Derivação normal) *Uma derivação sem fórmulas máximas e sem segmentos máximos é chamada de derivação normal. Uma derivação com essa característica está na forma normal.*

Ainda podem ocorrer redundâncias devido às regras $\vee$-*E* e Λ_C (na lógica de primeira ordem, a regra $\exists$-*E* também é incluída):

- $\vee$-*E* : uma aplicação dessa regra é dita redundante se nenhuma suposição em Σ_i é fechada pela regra $\vee$-*E* (ver tabela 4);
- Λ_C: uma aplicação dessa regra é dita redundante se nenhuma suposição em Σ é fechada em Σ' (ver tabela 4).

As *simplificações imediatas* removem esse tipo de redundância, conforme mostrado na tabela 4.

simplificação-$\vee$	**simplificação**-Λ_C
$\dfrac{\dfrac{\Sigma}{A_1 \vee A_2} \quad \dfrac{\Sigma_1}{B} \quad \dfrac{\Sigma_2}{B}}{B} \twoheadrightarrow \dfrac{\Sigma_i}{B}$	$\dfrac{\dfrac{\Sigma}{A} \quad \neg A}{[\Lambda]} \twoheadrightarrow \dfrac{\Sigma}{A}$ $\dfrac{\dfrac{\Sigma'}{\Lambda}}{A}$

Tabela 4: Simplificações imediatas

Definição 6 (derivação na forma normal completa) *Uma derivação na forma normal completa está na forma normal e não contém redundâncias causadas pelas regras $\vee$-E e Λ_C.*

2.4.2. A forma das derivações normais

As derivações que estão na forma normal possuem uma forma específica, a partir da qual algumas propriedades interessantes podem ser definidas. Na seção 2.4.5. é definida a *propriedade da subfórmula*. Um outro exemplo refere-se à última inferência de uma derivação normal na qual todas as suposições estão fechadas: a última inferência é sempre uma regra de introdução.

Uma derivação normal possui essencialmente duas partes: uma chamada *analítica* e a outra *sintética*. Na parte analítica, as suposições são "quebradas" em seus componentes através das regras de eliminação. enquanto que na parte sintética, os componentes são reunidos através de regras de introdução. Ainda existe uma parte chamada de *mínima*, composta por *fórmulas mínimas*, entre a parte analítica a parte sintética, na qual operações sobre fórmulas atômicas podem ocorrer. Em [Pra71a, Pra65], Prawitz fornece uma definição precisa da forma das derivações normais através das noções de *ramos* (do inglês, *branches*) e *caminhos* (do inglês, *paths*).

Uma derivação na forma normal completa pode ser transformada para a forma normal expandida, na qual todas as fórmulas *mínimas* são atômicas.

Definição 7 (derivação na forma normal expandida) *Uma derivação está na forma normal expandida se ela está na forma normal completa e todas as fórmulas mínimas são atômicas.*

As *expansões imediatas* transformam derivações na forma normal completa para a forma normal expandida. Na tabela 5, a expansão imediata para o conectivo $\wedge$ é mostrada.

$$\frac{\Sigma}{A \wedge B} \quad \twoheadrightarrow \quad \frac{\dfrac{\Sigma}{A \wedge B}\quad \dfrac{\Sigma}{A \wedge B}}{\dfrac{A}{A \wedge B}\quad \dfrac{B}{A \wedge B}}$$

Tabela 5: Expansão imediata

2.4.3. As reduções da regra do absurdo clássico

Ao considerar a normalização de provas, Prawitz restringe as aplicações da regra Λ_C à fórmulas atômicas. Dessa forma, é definido um conjunto de reduções da regra do absurdo clássico para atender a essa restrição. A partir dessas reduções, é assegurado (teorema 1, p. 39 de [Pra65]) que toda derivação para a lógica clássica em DN pode ser transformada para uma outra, na qual a conseqüência de toda aplicação da regra Λ_C é atômica.

redução-$\wedge$

$$\frac{\dfrac{[\neg(B \wedge C)]}{\dfrac{\Sigma}{\Lambda}}}{B \wedge C} \quad \twoheadrightarrow \quad \frac{\dfrac{\dfrac{\dfrac{\dfrac{B \wedge C^1}{B}\quad \neg B^2}{\Lambda}1}{[\neg(B \wedge C)]}}{\dfrac{\Sigma}{\Lambda}}2 \qquad \dfrac{\dfrac{\dfrac{\dfrac{B \wedge C^3}{C}\quad \neg C^4}{\Lambda}3}{[\neg(B \wedge C)]}}{\dfrac{\Sigma}{\Lambda}}4}{B \wedge C}$$

redução-$\longrightarrow$

$$\frac{\dfrac{[\neg(B \to C)]}{\dfrac{\Sigma}{\Lambda}}}{B \to C} \quad \twoheadrightarrow \quad \frac{\dfrac{\dfrac{\dfrac{B^1 \quad B \to C^2}{C}\quad \neg C^3}{\Lambda}2}{[\neg(B \to C)]}}{\dfrac{\dfrac{\Sigma}{\Lambda}3}{C}}1}{(B \to C)}$$

2.4.4. Os teoremas da forma normal e normalização

A partir das regras de *redução*, é possível *reduzir* uma derivação Π para uma outra. Dessa forma, o *procedimento de normalização* é definido. Para especificá-lo, as noções de *reducibilidade imediata*, *reducibilidade*, e *convertibilidade* são introduzidas [Pra71a].

Definição 8 (reducibilidade imediata) *Uma derivação* Π *reduz "imediatamente" para* Π' *se* Π' *é obtida a partir de* Π *substituindo-se uma subárvore de* Π *por uma redução aplicada a ela.*

Definição 9 (reducidibilidade) *Uma derivação* Π *reduz para* Π' *se existe uma sequência* $\Pi_1, \Pi_2, \ldots, \Pi_n, n \geq 1$*, onde* $\Pi_1 = \Pi$*;* Π_i *reduz imediatamente para* Π_{i+1}*, para cada* $i < n$*; e* $\Pi_n = \Pi'$*.*

Definição 10 (convertibilidade) *Uma derivação* Π_1 *"converte" para* Π_2 *se existe uma derivação* Π'*, de forma que* Π_1 *reduz para* Π' *e* Π_2 *é obtida a partir de* Π' *através de sucessivas simplificações imediatas ou expansões.*

Os teoremas da forma normal e da normalização conforme enunciados por Prawitz [Pra71a] são reportados a seguir.

Teorema 1 (forma normal) *Se* A *é derivável a partir de* Γ *em um dos sistemas* **M**, **I**, *ou* **C**, *então existe uma derivação normal (completa, expandida) de* A *a partir de* Γ *no sistema dado.*

Teorema 2 (normalização) *Toda derivação em* **M**, **I**, *ou* **C** *reduz (converte) para uma forma normal (completa, expandida).*

O teorema da forma normal assegura a existência da forma normal, ao passo que o teorema da normalização define um método para obter a formal normal de uma dada derivação.

A prova do teorema da normalização é feita por indução nos tamanhos da fórmula máxima e do segmento máximo.

Teorema 3 (normalização forte) *Toda derivação* Π *em* M, **I** *ou* C *reduz para uma única derivação* Π' *na forma normal e toda sequência de redução que começa em* Π *termina (em* Π'*).*

Prawitz utiliza a noção de *validade* na demonstração do teorema da normalização forte [Pra71a, Pra79]. A noção de *validade* [Pra79] corresponde à noção de *computabilidade* utilizada por Martin-Löf, e à noção de *réducibilité*, usada por Girard. Adicionalmente, uma noção similar já tinha sido utlizada por Tait em 1967: o conceito de *convertibilidade*.

Esse tipo de prova é chamada de prova semântica porque usa a propriedade de *validade* como uma ferramenta da seguinte forma: (i) primeiro prova-se que *validade* implica em normalização forte, *i.e.* qualquer sequência de redução que começa numa derivação válida termina; (ii) em seguida prova-se que *validade* é fechada sob reducibilidade. Em [Lei79], Leivant observa que pode ocorrer um problema nesse tipo de prova. A prova falha caso não se use a noção de *classes de suposições* no processo de fechar suposições. Existem também provas sintáticas. Em [Mas90] são encontradas provas semânticas e sintáticas para todos os conectivos na lógica clássica de primeira ordem.

2.4.5. Propriedade da subfórmula

Existem várias conseqüências e aplicações do teorema da normalização. Por exemplo, o seu uso em provas de consistência, o teorema da interpolação, etc. Aqui é enfatizado o *princípio da subfórmula*. Conforme analisado por Prawitz [Pra79] esse é o aspecto mais interessante do teorema da normalização, pois num certo sentido o teorema revela um conteúdo construtivo das derivações. O princípio da subfórmula é precisamente estabelecido como a seguir [Pra71a]:

Corolário 1 (princípio da subfórmula para M e I) *Cada fórmula numa derivação normal e completa de A a partir de Γ em* **M** *ou* **I** *é subfórmula de A ou de alguma fórmula de Γ.*

Corolário 2 (princípio da subfórmula em C) *Cada fórmula numa derivação normal de A a partir de Γ no sistema* **C***, ou é uma subfórmula de A; ou de alguma fórmula de Γ; ou uma suposição $\neg B$ fechada por Λ_C (no caso em que B é uma subfórmula do tipo mencionado ou uma ocorrência de Λ imediatamente abaixo da suposição).*

2.5. Conclusão

O trabalho de Prawitz representa um importante progresso para os sistemas de dedução natural. Entretanto, a sua formulação para a lógica clássica não inclui as constantes $\vee$ e $\exists$ como primitivas. Isso tem motivado várias pesquisas com o intuito de apresentar uma prova para os teoremas da normalização na lógica clássica completa, incluindo os conectivos $\vee$ e $\exists$ como primitivos.

Statman em [Sta74] foi quem primeiro apresentou um teorema da normalização para os sistemas de dedução natural na lógica clássica completa. Embora com métodos diferentes e mais simples, Stalmarck [Stal91] assim como L. Pereira e C. Massi [PerMas90] seguiram Statman em alguns detalhes técnicos e apresentaram uma solução alternativa para a lógica clássica completa. Esses trabalhos, juntamente com a proposta de Andou [And95], abordam o sistema de lógica clássica para DN conforme formalizado por Prawitz, i.e. através da inclusão da regra do absurdo clássico (Λ_C). Entretanto, os trabalhos de J. Seldin [Sel89] e J. von Plato [Pla98] apresentam os teoremas da forma normal para lógica clássica completa através do uso da *lei do terceiro excluído*, como preferia Gentzen.

3. Cálculo de Sequentes

O cálculo de seqüentes, proposto por Gentzen [Gen35], é um sistema dedutivo extremamente rico usado não somente para construir provas mas também para estudar as metapropriedades dos sistemas lógicos sob a perspectiva da Teoria da Prova [TH96]. Devido a certas assimetrias inerentes às regras da dedução natural, Gentzen propôs o cálculo de seqüentes com o intuito principal de demonstrar o *Hauptsatz*, também chamado de Teorema da Eliminação do Corte, para a lógica clássica de forma mais conveniente. Este teorema diz que toda prova pode ser transformada em uma prova normal, canônica, padrão. Conforme estudado na seção 2., Prawitz obteve resultados para os sistemas de dedução natural — os teoremas de normalização — análogos ao teorema da eliminação do corte para os sistemas de cálculo de seqüentes.

Os sistemas de cálculo de seqüentes e dedução natural possuem algumas vantagens em relação à apresentação axiomática tradicional. No sistema axiomático, há muitas possibilidades de escolha de axiomas para definir um dado símbolo lógico: qual deles é o melhor e qual critério usar para tal escolha? Há ainda problemas relacionados com o número de axiomas a considerar: quando um conjunto de axiomas é completo e quando é minimal? Mais ainda, no sistema axiomático a definição de um símbolo lógico é freqüentemente apresentada por um axioma onde outros símbolos lógicos aparecem. O sistema de Hilbert [Kle52], por exemplo, apresenta axiomas para definir o papel dedutivo de cada

símbolo lógico no qual a implicação é sempre utilizada. Gentzen foi capaz de apresentar o papel dedutivo de uma forma individualizada descartando o papel da implicação dos outros símbolos lógicos. Neste sentido, tanto no sistema de dedução natural quanto no de cálculo de seqüentes, as inferências são quebradas em passos atômicos, ou seja, cada regra de inferência trata de apenas uma constante lógica.

3.1. Os cálculos clássico e intuicionístico

São apresentados os cálculos clássico e intuicionístico para o fragmento proposicional definido sobre as constantes lógicas usuais: $\neg$ (negação), $\wedge$ (conjunção), $\vee$ (disjunção) e $\rightarrow$ (implicação). São usadas letras latinas maiúsculas $A, B, C, \ldots, A_1, B_1, C_1, \ldots, A_k, B_k, C_k, \ldots$ para designar fórmulas quaisquer e letras gregas maiúsculas $\Gamma, \Delta, \Gamma', \Delta', \ldots$ para designar seqüências arbitrárias de fórmulas do tipo $A_1, \ldots, A_n$, para $n \geq 0$.

A expressão $\Gamma \vdash \Delta$ é denominada *seqüente*. Se $\Gamma = A_1, A_2, \ldots, A_n$ e $\Delta = B_1, B_2, \ldots, B_m$, então o seqüente $\Gamma \vdash \Delta$ é usualmente interpretado como a fórmula $(A_1 \wedge A_2 \wedge \ldots \wedge A_n) \rightarrow (B_1 \vee B_2 \vee \ldots \vee B_m)$. Quando Γ e Δ são seqüências nulas de fórmulas, o seqüente $\vdash$ é chamado de *seqüente vazio* e é tomado como a representação do absurdo, da contradição.

Diz-se que A é conseqüência clássica (ou intuicionística) das hipóteses $A_1, \ldots, A_n$ sse existe uma prova para A no cálculo de seqüentes clássico (ou intuicionístico) a partir de tais hipóteses. Tal prova possui uma estrutura de árvore onde a raiz $A_1, \ldots, A_n \vdash A$ é denominada de *seqüente final*, as folhas são do tipo $B \vdash B$ e referenciadas por *seqüentes iniciais*, e cada *seqüente intermediário* é obtido por aplicação das regras de inferência do cálculo. Se $n = 0$, diz-se que A é um teorema clássico (ou intuicionístico). As regras do cálculo de seqüentes indicam como devem ser introduzidas as fórmulas à direita e à esquerda de $\vdash$.

3.1.1. As regras operacionais

As *regras operacionais* do cálculo de seqüentes são aquelas que dão o significado individualizado e operacional das constantes lógicas dizendo como usá-las. As regras operacionais de introdução à direita e à esquerda de $\vdash$ correspondem, em um certo sentido [GLT89], às regras de introdução e eliminação das constantes lógicas no sistema de dedução natural, respectivamente. Estas regras refletem o *Princípio da Composicionalidade* de Frege o qual afirma que o significado das fórmulas depende unicamente do significado de suas subfórmulas e do conectivo que as une. As regras operacionais são:

Conjunção

$$\frac{\Gamma, A \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\ L^1\wedge_1 \qquad \frac{\Gamma, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\ L\wedge_2$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma' \vdash B, \Delta'}{\Gamma, \Gamma' \vdash A \wedge B, \Delta, \Delta'}\ R^2\wedge$$

Disjunção

[1] L abrevia *Left*.
[2] R abrevia *Right*.

$$\frac{\Gamma, A \vdash \Delta \;\; \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \vee B \vdash \Delta, \Delta'} \; L\vee$$

$$\frac{\Gamma \vdash A, \Delta}{\Gamma \vdash A \vee B, \Delta} \; R\vee_1 \qquad \frac{\Gamma \vdash B, \Delta}{\Gamma \vdash A \vee B, \Delta} \; R\vee_2$$

Implicação

$$\frac{\Gamma \vdash A, \Delta \;\; \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \to B \vdash \Delta, \Delta'} \; L\to \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta} \; R\to$$

Negação

$$\frac{\Gamma \vdash A, \Delta}{\Gamma, \neg A \vdash \Delta} \; L\neg \qquad \frac{\Gamma, A \vdash \Delta}{\Gamma \vdash \neg A, \Delta} \; R\neg$$

A regra da conjunção pode ser entendida como: para provar $A \wedge B$ é preciso provar A e B, mas para refutar $A \wedge B$ basta refutar A ou refutar B. As outras regras têm interpretação semelhante.

3.1.2. As regras estruturais

Nos sistemas dedutivos em geral, algumas propriedades das derivações são explicitadas apenas a nível da metalinguagem. Por exemplo, em uma prova no sistema de dedução natural, é sempre possível permutar fórmulas, duplicá-las ou mesmo acrescentar novas hipóteses se certas restrições são obedecidas. No cálculo de seqüentes, entretanto, tais propriedades são explicitadas dentro do próprio cálculo através das chamadas *regras estruturais*. Neste sentido, diz-se que o cálculo de seqüentes é um meta-cálculo pois permite representar, a nível objeto, meta-propriedades das derivações.

Permutação[3]

$$\frac{\Gamma, A, B, \Gamma' \vdash \Delta}{\Gamma, B, A, \Gamma' \vdash \Delta} \; LX \qquad \frac{\Gamma \vdash \Delta, A, B, \Delta'}{\Gamma \vdash \Delta, B, A, \Delta'} \; RX$$

Enfraquecimento[4]

$$\frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta} \; LW \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash A, \Delta} \; RW$$

Contração

$$\frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta} \; LC \qquad \frac{\Gamma \vdash A, A, \Delta}{\Gamma \vdash A, \Delta} \; RC$$

A regra da permutação espelha a meta-propriedade de possibilitar trocar a ordem das fórmulas em uma prova. A regra do enfraquecimento à esquerda, por exemplo, representa a meta-propriedade de monotonicidade das lógicas dedutivas, ou seja, se Δ é provado a partir de um conjunto Γ de hipóteses, continuará sendo provado mesmo que seja acrescentada uma nova hipótese A à Γ. Em certo sentido, o "enfraquecimento" da prova se dá com o acréscimo de uma nova hipótese desnecessária. A regra de contração reflete a meta-propriedade que permite a utilização da mesma fórmula, em uma prova, mais de uma vez.

[3] *Exchange*, em inglês.
[4] *Weakening*, em inglês.

Observe que as regras estruturais independem das constantes lógicas da linguagem subjacente. Entretanto, tais regras determinam, de certa forma, o comportamento de tais símbolos. Por exemplo, se as vírgulas à esquerda do seqüente forem interpretadas como conjunções e aquelas à direita do seqüente como disjunções, as regras de permutação e contração poderiam ser entendidas como a expressão das leis de comutatividade e idempotência destes conectivos, respectivamente. Como outro exemplo, no cálculo de seqüentes intuicionístico (definido abaixo), embora as regras operacionais da negação sejam as mesmas clássicas, a ausência das regras estruturais RC e RX é suficiente para caracterizar indiretamente a negação intuicionística.

3.1.3. As regras de "identidade"

A primeira das *regras de identidade* reflete, na verdade, a propriedade reflexiva da relação de derivabilidade: qualquer fórmula é derivada dela mesma. Tal propriedade é expressa por um "axioma de identidade". Ele é essencial para qualquer prova no cálculo de seqüentes pois representam as folhas da árvore de prova, ou seja, o ponto de partida da prova. Observe que A não necessariamente é fórmula atômica.

$$A \vdash A \quad Identidade$$

A noção de composição de provas através de lemas, ou da transitividade da relação de derivabilidade, é também refletido a nível objeto através da regra do corte.

$$\frac{\Gamma \vdash A, \Delta \;\; \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'} \;\; Corte$$

Apesar do teorema da eliminação do corte, provado a seguir, demonstrar que tal regra é desnecessária ao sistema (embora não derivada das outras), isto não acarretará na perda da propriedade de transitividade da relação de derivabilidade, pois tal propriedade é facilmente provada no meta-nível, independente ou não de sua representação a nível objeto [Hac79].

3.1.4. O cálculo intuicionístico

Enquanto na dedução natural, a diferença entre os sistemas intuicionístico e clássico se reflete nas regras que definem a negação, é possível obter o cálculo de seqüentes intuicionístico a partir do clássico através de modificações na definição de seqüente. Por exemplo, pode-se impor uma restrição no número de fórmulas permitidas do lado direito do seqüente: no máximo uma (1) fórmula [Gen35]. Com esta restrição, o cálculo intuicionístico perde as regras estruturais: RC e RX. As demais regras são preservadas do cálculo clássico, a menos da regra $L\vee$ que é modificada para manter o mesmo contexto à direita do seqüente.

Disjunção intuicionística

$$\frac{\Gamma, A \vdash \Delta \;\; \Gamma', B \vdash \Delta}{\Gamma, \Gamma', A \vee B \vdash \Delta} \;\; L\vee_i$$

Outras possíveis definições do seqüente intuicionístico têm sido formuladas. Em [Masi93], por exemplo, é introduzida a noção de 2-seqüente e o cálculo intuicionístico é

obtido, como no caso acima, através de uma restrição no lado direto do 2-seqüente. Uma alternativa diferente é apresentada em [Dos85] para seqüentes de multi-níveis. Neste caso, o cálculo intuicionístico é obtido através do clássico pela restrição da aplicação da regra de enfraquecimento.

3.1.5. Exemplos

Os exemplos abaixo foram escolhidos de forma a ilustrar a aplicação de quase todas as regras. Todas as provas iniciam com axiomas de identidade. Exemplos de provas com o uso da regra do corte são apresentados durante a demontração do Teorema da Eliminação do Corte.

1. Exemplo de uso das regras LW e $R\rightarrow$:
$$\dfrac{\dfrac{\dfrac{A \vdash A}{A, B \vdash A}}{A \vdash B \rightarrow A}}{\vdash (A \rightarrow (B \rightarrow A)}$$
2. Exemplo de uso das regras RW, $L\rightarrow$ e RC :
$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B, A}}{\vdash A \rightarrow B, A} \quad A \vdash A}{(A \rightarrow B) \rightarrow A \vdash A, A}}{(A \rightarrow B) \rightarrow A \vdash A}}{\vdash ((A \rightarrow B) \rightarrow A) \rightarrow A}$$
3. Exemplo de uso das regras $L\neg$, $R\vee_2$, RX e $R\vee_1$:
$$\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{\vdash \neg A, A}}{\vdash A \vee (\neg A), A}}{\vdash A, A \vee (\neg A)}}{\vdash A \vee (\neg A), A \vee (\neg A)}}{\vdash A \vee (\neg A)}$$
4. Exemplo de uso das regras $R\neg$, $R\wedge$, $L\wedge_2$, LX, $L\wedge_1$ e LC :
$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{A, \neg A \vdash}}{A \vdash \neg\neg A} \quad \dfrac{\dfrac{B \vdash B}{B, \neg B \vdash}}{B \vdash \neg\neg B}}{A, B \vdash \neg\neg A \wedge \neg\neg B}}{A, A \wedge B \vdash \neg\neg A \wedge \neg\neg B}}{A \wedge B, A \vdash \neg\neg A \wedge \neg\neg B}}{A \wedge B, A \wedge B \vdash \neg\neg A \wedge \neg\neg B}}{A \wedge B \vdash \neg\neg A \wedge \neg\neg B}$$

3.2. O teorema da eliminação do corte

Esta demonstração evidencia a elegante simetria das regras do cálculo. Provas livre de corte são consideradas provas sem redundâncias, provas normais, embora sejam provas de tamanho consideravelmente maiores que as provas que usam lemas introduzidos pela regra do corte. Este teorema diz que: Toda derivação, seja ela clássica ou intuicionística, pode ser transformada em outra derivação, com o mesmo seqüente final, sem uso da regra do corte.

3.2.1. Demonstração

Embora a prova da eliminação do corte [Gen35, GLT89, Ung92] esteja aqui apresentada para o sistema clássico, para o caso intuicionístico a prova é análoga, *mutatis mutandis*. Esta prova é baseada em duas induções. Em uma delas, o objetivo é diminuir o grau das aplicações das regras do corte. Na outra indução, pretende-se diminuir a altura da subárvore de prova onde regra do corte é a última aplicada.

O *grau* $g(A)$ *de uma fórmula* é definido por:

1. $g(P) = 1$ onde P é fórmula atômica qualquer;
2. $g(\neg A) = g(A)+1$;
3. $g(A \vee B) = g(A \wedge B) = g(A \rightarrow B) = \max(g(A), g(B))+1$;

O *grau da aplicação de uma regra do corte* é definido pelo grau da fórmula eliminada pela regra.

O *grau* g *de uma prova* π é o grau maximal das aplicações das regras de corte na prova π, e $g(\pi) = 0$ sse π é livre de corte.

A *altura* $a(\pi)$ *de uma prova* é a altura de sua árvore associada: Se π termina com uma regra cujas premissas são provadas por $\pi_1, \ldots, \pi_n$ (n = 0, 1 ou 2) então $a(\pi) = max(a(\pi_1), \ldots, a(\pi_n)) + 1$ e max é uma função que fornece o número natural maximal do conjunto $\{a(\pi_1), \ldots, a(\pi_n)\}$.

A idéia central da prova da eliminação do corte é: (i) permutar o corte para cima diminuindo a altura da prova onde a regra do corte é a última regra aplicada, e (ii) reduzir a complexidade do corte diminuindo o seu grau. O processo de reescrita da eliminação do corte obedece os seguintes casos [Ung92] (as linhas duplas usadas abaixo indicam um número arbitrário de aplicação de regras estruturais):

1. Eliminação de cortes triviais. Por exemplo:
$$\frac{\Gamma \vdash A, \Delta \quad A \vdash A}{\Gamma \vdash A, \Delta}$$
é trocado por: $\Gamma \vdash A, \Delta$
2. Permutação dos cortes para cima da árvore de prova. Por exemplo:
$$\frac{\Gamma \vdash \Delta, A \qquad \dfrac{A, \Gamma' \vdash \Delta', B \quad A, \Gamma'' \vdash \Delta'', C}{A, \Gamma', \Gamma'' \vdash \Delta', \Delta'', B \wedge C}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta'', B \wedge C}$$
é trocado por:
$$\frac{\dfrac{\Gamma \vdash \Delta, A \quad A, \Gamma' \vdash \Delta', B}{\Gamma, \Gamma' \vdash \Delta, \Delta', B} \qquad \dfrac{\Gamma \vdash \Delta, A \quad A, \Gamma'' \vdash \Delta'', C}{\Gamma, \Gamma'' \vdash \Delta, \Delta'', C}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta'', B \wedge C}$$
3. Separação dos cortes. Por exemplo:
$$\frac{\Gamma \vdash \Delta, A \vee B \qquad \dfrac{A, \Gamma' \vdash \Delta' \quad B, A \vee B, \Gamma'' \vdash \Delta''}{A \vee B, A \vee B, \Gamma', \Gamma'' \vdash \Delta', \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$
é trocado por:
$$\frac{\Gamma \vdash \Delta, A \vee B \qquad \dfrac{\Gamma \vdash \Delta, A \vee B \qquad \dfrac{A, \Gamma' \vdash \Delta' \quad B, A \vee B, \Gamma'' \vdash \Delta''}{A \vee B, A \vee B, \Gamma', \Gamma'' \vdash \Delta', \Delta''}}{A \vee B, \Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

4. Eliminação dos cortes os quais possuem uma premissa introduzida pela regra de enfraquecimento. Por exemplo:

$$\dfrac{\dfrac{\Gamma \vdash \Delta}{\Gamma \vdash A, \Delta} \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

é trocado por:

$$\dfrac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

5. Redução da complexidade dos cortes:
   (a) $R\wedge$ e $L\wedge_1$

$$\dfrac{\dfrac{\Gamma \vdash A, \Delta \quad \Gamma' \vdash B, \Delta'}{\Gamma, \Gamma' \vdash A \wedge B, \Delta, \Delta'} \quad \dfrac{\Gamma'', A \vdash \Delta''}{\Gamma'', A \wedge B \vdash \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

   é trocado por

$$\dfrac{\dfrac{\Gamma \vdash A, \Delta \quad \Gamma'', A \vdash \Delta''}{\Gamma, \Gamma'' \vdash \Delta, \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

   (b) $R\wedge$ e $L\wedge_2$

$$\dfrac{\dfrac{\Gamma \vdash A, \Delta \quad \Gamma' \vdash B, \Delta'}{\Gamma, \Gamma' \vdash A \wedge B, \Delta, \Delta'} \quad \dfrac{\Gamma'', B \vdash \Delta}{\Gamma'', A \wedge B \vdash \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

   é trocado por

$$\dfrac{\dfrac{\Gamma' \vdash B, \Delta' \quad \Gamma'', B \vdash \Delta''}{\Gamma', \Gamma'' \vdash \Delta', \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

   (c) $R\vee_1$ e $L\vee$

$$\dfrac{\dfrac{\Gamma \vdash A, \Delta}{\Gamma \vdash A \vee B, \Delta} \quad \dfrac{\Gamma', A \vdash \Delta' \quad \Gamma'', B \vdash \Delta''}{\Gamma', \Gamma'', A \vee B \vdash \Delta', \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

   é trocado por

$$\dfrac{\dfrac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

   (d) $R\vee_2$ e $L\vee$

$$\dfrac{\dfrac{\Gamma \vdash B, \Delta}{\Gamma \vdash A \vee B, \Delta} \quad \dfrac{\Gamma', A \vdash \Delta' \quad \Gamma'', B \vdash \Delta''}{\Gamma', \Gamma'', A \vee B \vdash \Delta', \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

   é trocado por

$$\dfrac{\dfrac{\Gamma \vdash B, \Delta \quad \Gamma'', B \vdash \Delta''}{\Gamma, \Gamma'' \vdash \Delta, \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

   (e) $R\neg$ e $L\neg$

$$\dfrac{\dfrac{\Gamma, A \vdash \Delta}{\Gamma \vdash \neg A, \Delta} \quad \dfrac{\Gamma' \vdash A, \Delta'}{\Gamma', \neg A \vdash \Delta'}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

é trocado por

$$\dfrac{\dfrac{\Gamma' \vdash A, \Delta' \quad \Gamma, A \vdash \Delta}{\Gamma', \Gamma \vdash \Delta', \Delta}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

(f) $R \rightarrow$ e $L \rightarrow$

$$\dfrac{\dfrac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \rightarrow B, \Delta} \quad \dfrac{\Gamma' \vdash A, \Delta' \quad \Gamma'', B \vdash \Delta''}{\Gamma', \Gamma'', A \rightarrow B \vdash \Delta', \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

é trocado por

$$\dfrac{\dfrac{\dfrac{\Gamma' \vdash A, \Delta' \quad \Gamma, A \vdash B, \Delta}{\Gamma', \Gamma \vdash \Delta', B, \Delta}}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'} \qquad \Gamma'', B \vdash \Delta''}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

3.2.2. Conseqüências da eliminação do corte

Uma conseqüência trivial do teorema da eliminação do corte é a prova da (auto)consistência do sistema lógico sem precisar apelar para a definição de um modelo que o seja equivalente. A regra de corte é, de fato, necessária para provar o seqüente vazio, derivável de qualquer seqüente contraditório $\vdash A \wedge \neg A$:

$$\dfrac{\vdash A \wedge \neg A \qquad \dfrac{A \wedge \neg A, \dfrac{A \wedge \neg A, \dfrac{\dfrac{\dfrac{A \vdash A}{A, \neg A \vdash}}{A \wedge \neg A, A \vdash}}{A \wedge \neg A \vdash}}{A \wedge \neg A, A \wedge \neg A \vdash}}{\vdash}$$

Uma outra conseqüência importante é a *propriedade da subfórmula*: Com exceção da regra do corte, todas as outras regras no cálculo de seqüentes têm a propriedade de que as premissas são subfórmulas da conclusão. Em particular, uma prova sem corte usa somente subfórmulas da fórmula do seqüente final. Tal previsibilidade torna possível o desenvolvimento de eficientes provadores automáticos de teoremas.

Na definição de um sistema lógico formal, é usual requerer que a definição das constantes lógicas sejam conservativas, ou seja, que elas sejam inseridas e retiradas numa prova sem perda de informação. Esta propriedade pode ser expressa através do *princípio da inversão*: As regras de introdução à esquerda das constantes lógicas devem ser regras inversas das regras de introdução à direita. O processo de reescrita subjacente ao teorema da eliminação do corte torna explícito a interação entre as regras da esquerda e da direita refletindo as profundas simetrias entre elas e assegurando o princípio da inversão [GLT89].

Um exemplo de um operador não conservativo é o famoso conectivo ‘tonk’ proposto por Prior [Pri60] e definido como:

$$\dfrac{\Gamma, B \vdash \Delta}{\Gamma, A \; tonk \; B \vdash \Delta} \; L_{tonk} \qquad \dfrac{\Gamma \vdash A, \Delta}{\Gamma \vdash A \; tonk \; B, \Delta} \; R_{tonk}$$

Considere, agora, a seguinte prova em um sistema lógico que possui as regras para o conectivo $tonk$ e que possui a regra do corte:

$$\dfrac{\dfrac{A \vdash A}{A \vdash A\ tonk\ B} \qquad \dfrac{B \vdash B}{A\ tonk\ B \vdash B}}{A \vdash B}$$

Vê-se que este sistema lógico é trivial já que ele permite a prova de qualquer fórmula B a partir de uma fórmula A arbitrária. Por outro lado, a regra do corte neste sistema não é dispensável (a eliminação do corte não é provada) uma vez que o conjunto de conseqüências do sistema lógico com a regra do corte é diferente do conjunto de conseqüências do sistema sem corte. Em certo sentido, o teorema da eliminação do corte funciona como um teste para saber se as constantes lógicas foram bem definidas.

Sempre que se define um novo conceito, é importante que critérios de equivalência sejam estabelecidos. Por exemplo, como caracterizar uma equivalência para provas em cálculo de seqüentes? A grosso modo, poderia se dizer que duas provas do mesmo seqüente final são equivalentes se elas fossem reduzidas a uma mesma forma normal, ou seja, a uma mesma prova livre de corte. Entretanto, para um mesmo seqüente final, é possível se identificar uma classe não unitária de provas livres de corte. Como definir um "isomorfismo" entre os elementos desta classe? Este critério de equivalência entre provas no cálculo de seqüentes é melhor investigado no âmbito da teoria da prova para lógica linear, através do conceito de redes de prova (veja seção 4.).

Embora os resultados da eliminação do corte, citados acima, possuam todos uma característica positiva, uma conseqüência negativa de tal teorema é a expansão exponencial da prova ao fim do processo de reescrita da mesma. Por exemplo, nos casos 2. e 3. da seção 3.2.1., há uma duplicação da árvore de prova quando se busca "empurrar" corte para a cima ou mesmo "separar" os cortes. Um modelo combinatório para explicar a expansão das provas através de uma perspectiva geométrica é estudado em [Car99].

3.3. Conclusão

Foi apresentado, nesta seção, os sistemas clássico e intuicionístico de cálculo de seqüentes, como também demonstrado o teorema da eliminação do corte e discutido suas diversas conseqüências.

O teorema da eliminação do corte permite várias aplicações para cientistas da computação. O processo de reescrita da prova de eliminação do corte tem estreita ligação com a estratégia de resolução adotada em diversos provadores automáticos de teoremas. A linguagem de programação PROLOG é, na verdade, uma implementação de um fragmento do cálculo de seqüentes clássico com adicionais controles inerentes à programação. Os *tableaux* sinalizados, por outro lado, são também casos especiais do cálculo de seqüentes [GLT89, TH96].

Os sistemas de cálculo de seqüentes têm sido propostos para outras lógicas, além da lógica clássica e intuicionística, a saber: linear (veja seção 4.), modal, paraconsistente [Mart97, MP94], não-monotônica [Mart97, MPP99], entre outras. Têm sido pesquisados sistemas com fórmulas em apenas um lado do símbolo de seqüente, assim como investigados resultados relacionados ao princípio da inversão, aos teoremas da interpolação e da separação, e à expansão das provas após a eliminação do corte. Veja, em [TH96], uma rica referência sobre todas estas possíveis linhas de pesquisa.

4. Lógica Linear

A lógica linear, introduzida por Girard em [Gir87a], se tornou bastante popular na comunidade de ciência da computação. A grande novidade é a existência de novos conectivos que formam um novo sistema lógico com várias características interessantes, como por exemplo a possibilidade de se interpretar um seqüente como um estado de um sistema, assim como o tratamento de uma fórmula como um recurso. Um outro aspecto considerado na literatura como bastante inovativo é a noção de *redes-de-prova* (do inglês, *proof-nets*). A idéia é fornecer uma representação "grafo-teórica" para as deduções na lógica linear. Girard define *estruturas-de-prova*, *i.e.* grafos de prova, chamados por ele de "a dedução natural do cálculo de seqüentes linear" [Gir87a], utilizando a noção de *links*, que são relações entre ocorrências de fórmulas. Aquelas *estruturas-de-prova* que representam provas logicamente corretas são chamadas de *redes-de-prova*. Um dos aspectos talvez mais interessantes da teoria de redes-de-prova é a possibilidade de caracterizar dentro da classe de estruturas-de-prova a subclasse de *redes-de-prova* através de critérios puramente "*geométricos/algébricos*". *i.e.* baseado apenas na estrutura dos grafos de prova. Aqui são apresentados dois desses critérios, a proposta de Danos e Regnier [DanReg89] e o critério definido por Asperti e Dore [Asp95], que se baseia no paradigma "provas-como-processos", definido por Abramsky em [Abr94].

4.1. Eliminação das regras estruturais

Na lógica linear as fórmulas são vistas como um recurso, diferentemente do que acontece na lógica clássica, onde as proposições são verdades estáveis que podem ser usadas várias vezes em uma prova. Dessa forma, as regras estruturais *enfraquecimento* e *contração* são eliminadas. Enquanto a primeira permitiria o descarte de um recurso a segunda permitiria a sua duplicação. Na lógica linear, esse tipo de operação só pode ser feita de forma controlada através dos operadores exponenciais.

A regra de *permutação* não é eliminada. Ela expressa a propriedade de comutatividade dos multiplicativos.

4.2. Os conectivos aditivos e multiplicativos

Na lógica clássica, os conectivos $\wedge$ e $\vee$ podem possuir diferentes formulações de suas regras no cálculo de seqüentes. Por exemplo, a regra da direita do $\wedge$ pode possuir as seguintes variações:

Exemplo 8 *A apresentação mais à esquerda do conectivo $\wedge$ é* **aditiva**, *pois o contexto (Δ) é compartilhado, ao passo que a regra mais à direita é* **multiplicativa**, *o contexto (Δ, Δ') não é compartilhado.*

$$\frac{\Gamma \vdash \Delta, A \qquad \Gamma \vdash \Delta, B}{\Gamma \vdash \Delta, A \wedge B} \wedge -right - Ad \qquad \frac{\Gamma \vdash \Delta, A \qquad \Gamma' \vdash \Delta', B}{\Gamma, \Gamma' \vdash \Delta, \Delta', A \wedge B} \wedge -right - M$$

Essas duas formulações na lógica clássica são equivalentes por causa da existência de regras estruturais. Entretanto, na lógica linear, devido à falta de regras estruturais são introduzidas duas conjunções e duas disjunções. Dessa forma, tem-se os conectivos multiplicativos: conjunção ($\otimes$) e disjunção ($\wp$); e os conectivos aditivos: conjunção (&) e disjunção ($\oplus$).

4.3. Os operadores exponenciais

Para aumentar o poder de expressão da lógica linear, as regras estruturais são reintroduzidas, porém de forma controlada através dos operadores exponenciais: !(*of course*) e ?(*why not*), conforme mostrado na tabela 7.

4.4. A negação linear

Girard argumenta que através da negação linear é possível se ter a "dupla negação" sem perder "construtividade". A negação linear é introduzida através de suas regras, mostradas na tabela 7, e de um conjunto de equações que expressam as dualidades de "De Morgan", mostrado na tabela 6 (aqui apenas o fragmento proposicional é apresentado).

$1^\perp \equiv \perp$	$\perp^\perp \equiv 1$
$\top^\perp \equiv 0$	$0^\perp \equiv \top$
$A \equiv A^{\perp\perp}$	
$(A \otimes B)^\perp \equiv A^\perp \wp B^\perp$	$(A \wp B)^\perp \equiv A^\perp \otimes B^\perp$
$(A \& B)^\perp \equiv A^\perp \oplus B^\perp$	$(A \oplus B)^\perp \equiv A^\perp \& B^\perp$
$(A \multimap B) \equiv A^\perp \wp B$	
$(!A)^\perp \equiv ?A^\perp$	$(?A)^\perp \equiv !A^\perp$

Tabela 6: Equações "De Morgan"

4.5. O cálculo de seqüentes linear

As regras do cálculo de seqüentes linear são mostradas na tabela 7. As constantes $\top$ (verdade) e $\perp$ (falsidade) assumem as suas versões multiplicativas e aditivas:

- multiplicativa: 1 para o valor "verdade" e $\perp$ para o "falso";
- aditiva: $\top$ para o valor "verdade" e 0 para o "falso".

A implicação linear é um conectivo definido, conforme mostrado na tabela 6.

Exemplo 9 *Assim como o cálculo de seqüentes de Gentzen, o cálculo de seqüentes linear permite diferentes provas do mesmo seqüente. O seqüente* $\vdash (A \otimes B) \otimes C, A^\perp \wp B^\perp, C^\perp$ *possui as seguintes provas:*

$$\dfrac{\dfrac{\dfrac{\vdash A, A^\perp \qquad \vdash B, B^\perp}{\vdash A \otimes B, A^\perp, B^\perp}\otimes}{\vdash A \otimes B, A^\perp \wp B^\perp}\wp \qquad \vdash C, C^\perp}{\vdash (A \otimes B) \otimes C, A^\perp \wp B^\perp, C^\perp}\otimes$$

e

$$\dfrac{\dfrac{\dfrac{\vdash A, A^\perp \qquad \vdash B, B^\perp}{\vdash A \otimes B, A^\perp, B^\perp}\otimes \qquad \vdash C, C^\perp}{\vdash (A \otimes B) \otimes C, A^\perp, B^\perp, C^\perp}\otimes}{\vdash (A \otimes B) \otimes C, A^\perp \wp B^\perp, C^\perp}\wp$$

$\dfrac{}{A \vdash A}\ Identity$	$\dfrac{\Gamma' \vdash \Delta', A \qquad A, \Gamma \vdash \Delta}{\Gamma', \Gamma \vdash \Delta, \Delta'} Cut$
$\dfrac{\Gamma \vdash \Delta}{\Gamma' \vdash \Delta'}\ (exchange)$	
$\dfrac{}{\vdash 1}\ (one)$	$\dfrac{\vdash \Gamma}{\vdash \Gamma, \bot}\ (false)$
$\dfrac{\Gamma \vdash A, \Delta \qquad \Gamma' \vdash B, \Delta'}{\Gamma, \Gamma' \vdash A \otimes B, \Delta, \Delta'} \otimes - right$	$\dfrac{\Gamma, A, B \vdash \Delta}{\Gamma, A \otimes B \vdash \Delta} \otimes - left$
$\dfrac{\Gamma \vdash A, B, \Delta}{\Gamma \vdash A \wp B, \Delta} \wp - right$	$\dfrac{\Gamma, A \vdash \Delta \qquad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \wp B \vdash \Delta, \Delta'} \wp - left$
$\dfrac{}{\vdash \Gamma, \top}\ (true)$	*(no rule for zero)*
$\dfrac{\Gamma \vdash A, \Delta \qquad \Gamma \vdash B, \Delta}{\Gamma \vdash A \& B, \Delta} \& - right$	
$\dfrac{\Gamma, A \vdash \Delta}{\Gamma, A \& B \vdash \Delta} \& - left_1$	$\dfrac{\Gamma, B \vdash \Delta}{\Gamma, A \& B \vdash \Delta} \& - left_2$
$\dfrac{\Gamma \vdash A, \Delta}{\Gamma \vdash A \oplus B, \Delta} \oplus - right_1$	$\dfrac{\Gamma \vdash B, \Delta}{\Gamma \vdash A \oplus B, \Delta} \oplus - right_2$
$\dfrac{\Gamma, A \vdash \Delta \qquad \Gamma, B \vdash \Delta}{\Gamma, A \oplus B \vdash \Delta} \oplus - left$	
$\dfrac{\Gamma, A \vdash \Delta}{\Gamma \vdash A^{\bot}, \Delta} \bot - left$	
$\dfrac{\Gamma, !A, !A \vdash \Delta}{\Gamma, !A \vdash \Delta} !Contraction$	$\dfrac{\Gamma \vdash \Delta}{\Gamma, !A \vdash \Delta} !Weakening$
$\dfrac{\Gamma, A \vdash \Delta}{\Gamma, !A \vdash \Delta} !Dereliction$	$\dfrac{!\Gamma \vdash A, ?\Delta}{!\Gamma \vdash !A, ?\Delta} Ofcourse!$
$\dfrac{\Gamma, ?A, ?A \vdash \Delta}{\Gamma, ?A \vdash \Delta} ?Contraction$	$\dfrac{\Gamma \vdash \Delta}{\Gamma, ?A \vdash \Delta} ?Weakening$
$\dfrac{\Gamma \vdash A, \Delta}{\Gamma \vdash ?A, \Delta} ?Dereliction$	$\dfrac{!\Gamma, A \vdash ?\Delta}{!\Gamma, ?A \vdash ?\Delta} Whynot?$

Tabela 7: Cálculo de seqüentes linear proposicional

4.6. O fragmento multiplicativo

Devido ao grande número de conectivos na lógica linear, a sua divisão em fragmentos se tornou importante. Por falta de espaço, aqui será apresentado apenas o fragmento multiplicativo (MLL), principalmente por ser o fragmento mais utilizado em aplicações em computação como também por ser mais simples.

O fragmento multiplicativo é composto do axioma da identidade; das regras de inferência para os conectivos multiplicativos; da permutação e corte; e das equações de negação referentes à negação linear e aos conectivos multiplicativos. A seguir é apresentado o significado intuitivo de cada conectivo do fragmento multiplicativo [Cav01].

Implicação linear ($\multimap$) . Desde que na lógica linear uma fórmula é vista como um recurso, ao deduzirmos B a partir de A e $A \multimap B$, não temos mais A, por exemplo. A fírmula A foi consumida no processo de produção de B e não pode ser usada novamente. A implicação linear é vista como uma função linear, na qual os argumentos são usados uma única vez.

Times ($\otimes$) . A conjunção multiplicativa também é diferente da lógica clássica. $A \otimes A$ é diferente de A, pois $A \otimes A$ significa que temos duas cópias de A.

Par ($\wp$) . Por ser um conectivo com um significado pouco intuitivo, ele é dado através da equação $A \wp B \equiv A^{\perp} \multimap B$.

Of course (!) . Uma fórmula com a modalidade *of course* $!A$ equivale a $A \otimes \ldots \otimes A$, ou seja temos um número desejado de cópias de A.

Why not (?) . Essa modalidade é definida pela equação $?A \equiv A \wp \ldots \wp A$, ou seja, é possível se ter um número desejado de cópias do lado direito do seqüente.

4.7. Redes-de-prova (*"proof-nets"*)

As *redes-de-prova* (do inglês, *proof-nets*) foram introduzidas por Girard em seu artigo [Gir87a], no qual é feita a apresentação da lógica linear. A idéia é introduzir uma representação grafo-teórica para deduções na lógica linear.

As *redes-de-prova* representam a principal ferramenta para o propósito de Girard em estudar o que ele chama de "*geometria das deduções*". O estudo da chamada "geometria das deduções" se refere a uma forma particular de se utilizar os grafos de fluxo ocorrências de fórmulas das provas na exploração das simetrias de um sistema de prova. Na seção 4.7.3. são apresentados *critérios de corretude* para as *redes-de-prova*, baseados apenas em uma "análise puramente geométrica" da estrutura dos grafos que representam provas. Talvez esse seja o aspecto mais interessante da teoria das *redes-de-prova*. Por razões didáticas, aqui será apresentado apenas o fragmento multiplicativo sem constantes (MLL^{-}).

Girard defende seu programa através de uma série de críticas feitas à dedução natural e ao cálculo de seqüentes. Para ele, as *redes-de-provas*, chamadas de "a dedução natural para o cálculo de seqüentes linear", reunem as boas características desses dois cálculos [Gir87a]. Através da noção de *links*, i.e. uma relação entre ocorrências de fórmula, Girard define o conceito de *estruturas-de-prova*, que são os grafos de prova. Aquelas *estruturas-de-prova* que são logicamente corretas são chamadas de *redes-de-prova*.

4.7.1. Definição dos links

Dentro do programa de estudar a "geometria das deduções" Girard propõe o conceito de *redes-de-prova*. A idéia é substituir uma dedução em DN $\begin{array}{c}\Gamma\\ \vdots\\ A\end{array}$ por uma *rede-de-prova* com várias conclusões $\Gamma^{\perp}, A$, usando o símbolo de negação ($\perp$) sempre que uma fórmula é "invertida" de posição, ou seja, uma hipótese ou premissa passa a ser uma conclusão, e vice-versa [Gir89]. São definidos os *links* de identidade: *link axioma* e *link cut*, que permitem a substituição de uma hipótese por uma conclusão e vice-versa.

Figura 1: Links identidade

Mais dois *links* são definidos a seguir.

Link par

A seguinte inferência em DN:

$$\dfrac{\begin{array}{c}[A]\\ \vdots\\ B\end{array}}{A \to B}$$

é substituída por

$$\dfrac{A^{\perp} \qquad B}{A^{\perp} \wp B}$$

Consequentemente a seguinte regra binária (*link par*) é obtida:

$$\dfrac{C \qquad D}{C \wp D}$$

As fórmulas C e D são premissas do *link*, enquanto que $C \wp D$ é a conclusão.

Link times

A tradicional regra para eliminação do $\rightarrow$ em DN:

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

é substituída pela inferência mostrada na figura 2. A partir dessa figura, o *link times* é definido:

$$\frac{C \qquad D}{C \otimes D}$$

As fórmulas C e D são premissas do *link*, ao passo que $C \otimes D$ é a conclusão.

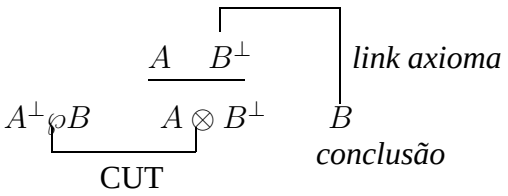

Figura 2: Modus ponens

Observa-se que as fórmulas B e $A \rightarrow B$ (*i.e.* $A^{\perp} \wp B$) são invertidas na figura 2.

4.7.2. Estruturas-de-prova

Através dos *links* definidos anteriormente, é possível construir *estruturas-de-prova*, que possuem uma notação similar à dedução natural, entretanto são construídas com várias conclusões, como no cálculo de seqüentes.

Definição 11 (estruturas-de-prova) *Uma estrutura-de-prova para o fragmento MLL^{-} consiste de (i) um conjunto não vazio de ocorrências de fórmulas (deve existir pelo menos uma ocorrência de fórmula) juntamente com (ii) um conjunto de* links *entre as ocorrências de fórmulas. Esses links foram definidos na seção 4.7.1.:* links axioma, link cut, link par *e o* link times*. Além disso, (i) e (ii) devem satisfazer os seguintes requerimentos: (a) toda ocorrência de fórmula da estrutura deve ser conclusão de um único* link*; (b) toda ocorrência de fórmula deve ser premissa de no máximo um* link*.*

As *estruturas-de-prova* podem ser naturalmente definidas em termos de grafos conexos não orientados cujos vértices são rotulados com ocorrências de fórmulas e cujas arestas são definidas a partir dos *links*:

- *links identidade*: existe uma aresta entre as duas fórmulas do *link*;
- *link par/times*: existe uma aresta partindo de cada uma das premissas para a conclusão.

A correspondência entre uma prova de Γ no cálculo de seqüentes linear e uma *estrutura-de-prova* cujas conclusões são as fórmulas de Γ é dada a partir do seguinte procedimento:

1. o *axioma identidade* é associado com o *link axioma*:

$$\vdash A^{\perp},\ A \quad \twoheadrightarrow \quad \overline{A \quad A^{\perp}}$$

2. A *regra par* é associada com o *link par*:

$$\dfrac{\begin{array}{c}\Pi \\ \vdots \\ \vdash \Gamma, A, B\end{array}}{\vdash \Gamma, A \wp B} \quad \twoheadrightarrow \quad \begin{array}{c} PS \\ \Gamma \\ \overline{A \quad B} \\ A \wp B \end{array}$$

onde PS é uma *estrutura-de-prova* correspondente à prova Π.

3. $$\dfrac{\begin{array}{cc}\Pi_1 & \Pi_2 \\ \vdots & \vdots \\ \vdash \Gamma, A & \vdash B, \Delta\end{array}}{\vdash \Gamma, \Delta, A \otimes B} \quad \twoheadrightarrow \quad \begin{array}{ccc} PS_1 & & PS_2 \\ \Gamma & & \Delta \\ & \overline{A \quad B} & \\ & A \otimes B & \end{array}$$

onde PS_1 e PS_2 são *estruturas-de-prova* associadas à Π_1 e Π_2, respectivamente.

Exemplo 10 *A estrutura-de-prova mostrada na figura 3 é construída a partir das deduções no cálculo de seqüentes linear mostradas no exemplo 9.*

Observa-se que uma única estrutura-de-prova corresponde a duas provas no cálculo de seqüentes linear. Esse exemplo ilustra uma característica importante das redes-de-prova: unicidade.

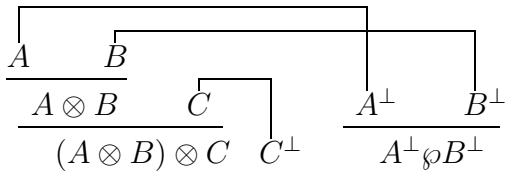

Figura 3: Estrutura-de-prova

A *estrutura-de-prova* mostrada nesse exemplo é uma *rede-de-prova*, como poderá ser verificado através da aplicação do critério global de corretude. Entretanto, a *estrutura-de-prova* mostrada na figura 4 não é uma *rede-de-prova*.

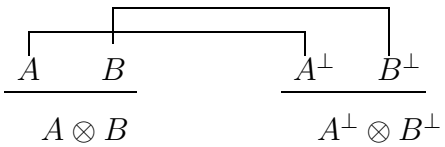

Figura 4: Estrutura-de-prova

Exemplo 11 *Um outro exemplo de uma estrutura-de-prova que também é uma* rede-de-prova *é mostrado na figura 5.*

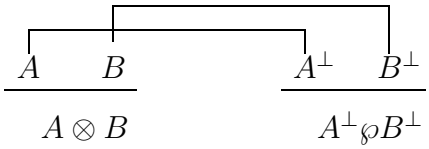

Figura 5: Estrutura-de-prova

4.7.3. Os critérios de corretude

Talvez a característica mais interessante das *redes-de-prova* seja a definição de um critério puramente geométrico para caracterização da classe de *estruturas-de-prova* que representam provas corretas. Existem vários critérios de corretude, por exemplo: (i) o critério conhecido como '*No shorttrip condition*': introduzido originalmente por Girard em [Gir87a]; (ii) O critério de Danos e Regnier: é uma simplificação do critério original de Girard [DanReg89]; (iii) o critério de Asperti: é uma generalização da condição '*no shorttrip*' dada por Girard, onde as viagens ('*trips*') são processos distribuídos, em oposição aos processos sequencias como originalmente definido [Asp95]. Aqui são introduzidos brevemente os critérios (ii) e (iii).

O critério de Danos-Regnier . Esse critério consiste em associar um conjunto de grafos a uma *estrutura-de-prova*. Se cada grafo desse conjunto é conexo e acíclico, ou seja é uma árvore, então a *estrutura-de-prova* é uma *rede-de-prova*.

A partir da definição de *estruturas-de-prova* em termos de grafos a noção de *grafos D-R* é introduzida como a seguir:

Definição 12 (grafos D-R) *Dada uma estrutura-de-prova P definida em termos de grafos, um grafo D-R G associado à P é um subgrafo de P obtido a partir da remoção de exatamente uma das duas arestas (*i.e. $(A, A\wp B)$, $(B, A\wp B)$*) de* cada *link par de P.*

A partir dessa definição a classe de *estruturas-de-prova* que são *redes-de-prova* é estabelecida:

Definição 13 (rede-de-prova) *Uma estrutura-de-prova P é uma rede-de-prova se e somente se todo grafo D-R associado à P é acíclico e conexo.*

Exemplo 12 *Os grafos D-R associados à estrutura-de-prova da figura 5 são mostrados na figura 6. Eles são acíclicos e conexos, portanto a estrutura-de-prova é uma rede-de-prova.*

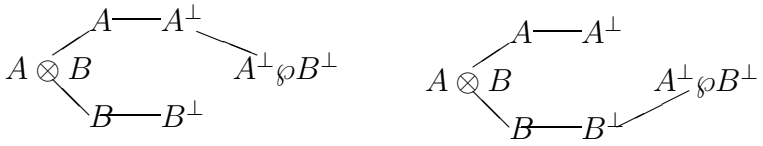

Figura 6: Grafos D-R

Exemplo 13 *O grafo D-R associado à estrutura-de-prova mostrada na figura 4 é cíclico, conforme ilustrado na figura 7.*

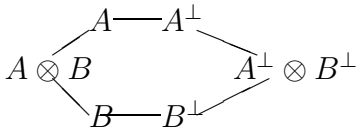

Figura 7: Grafo D-R

O critério baseado no paradigma "provas-como-processos" . O critério de Asperti [Asp95, AsDo94] é baseado no paradigma "provas-como-processos", introduzido por S. Abramsky [Abr94]. Nessa abordagem, as fórmulas sã vistas como processos distribuídos, o fluxo de informação nas estruturas-de-prova corresponde à computação do sistema e as *redes-de-prova* são definidas como *estruturas-de-prova* livres de '*deadlock*'.

O critério proposto por Asperti se aplica ao fragmento multiplicativo da lógica linear (MLL) com a regra mix[5] (esse fragmento é chamado de '*direct logic*'). O símbolo "$\|$" (composição paralela) é utilizado para denotar o conectivo "$\wp$" (par). Na prova de corretude do critério a condição de aciclicidade de Danos e Regnier é utilizada: *Uma estrutura-de-prova P é uma rede-de-prova se e somente se todos os grafos D-R associados a P são acíclicos.* A condição de conectividade não é utilizada aqui.

O mecanismo para verificar que classe de estruturas-de-prova são livres de '*deadlock*' é baseado no fluxo de informação do sistema:

- Inicialmente todos os processos finais estão ativos (os processos finais correspondem às conclusões);
- o *link axioma* representa *sincronização*. Os processos (fórmulas) A e $A^\perp$ conectados através do *link axioma* terminam se ambos estão ativos;
- o *link times* representa *exclusão mútua*. A ativação de $A \otimes B$ produz primeiro a ativação de A ou B. Depois que o primeiro subprocesso escolhido termina, o outro processo é ativado. A escolha de qual subprocesso será primeiro ativado é não-determinística. O processo $A \otimes B$ termina quando o segundo subprocesso termina.

Definição 14 (estrutura-de-prova livre de 'deadlock') *Uma estrutura-de-prova é livre de deadlock se para qualquer escolha nos* links $\otimes$*, todos os processos finais eventualmente terminam.*

Asperti prova que as *redes-de-prova* são justamente as *estruturas-de-prova* que são livres de '*deadlock*'. Alguns exemplos de [AsDo94] são reportados aqui.

Exemplo 14 *A estrutura-de-prova mostrada na figura 4 não é livre de* 'deadlock'*. Inicialmente, os dois processos $A \otimes B$ e $A^\perp \otimes B^\perp$ estão ativos. Se o primeiro processo decide ativar o subprocesso A, e o segundo processo decide ativar o subprocesso $B^\perp$, nenhum outro movimento pode ser feito.*

Exemplo 15 *A estrutura-de-prova mostrada na figura 8 é livre de* 'deadlock'*. Inicialmente, os processos ativos são $A \otimes B$, C, $A^\perp \otimes C^\perp$ e $B^\perp$. Para verificar que o sistema é livre de* 'deadlock *é preciso considerar todas as possíveis escolhas nos* links $\otimes$*. Existem*

[5] $\dfrac{\vdash \Gamma \quad \vdash \Delta}{\vdash \Gamma, \Delta} mix$

quatro casos. Aqui será considerado apenas um caso. Supondo que $A \otimes B$ começa ativando A e $A^{\perp} \otimes C^{\perp}$ ativando $C^{\perp}$. Desde que C e $C^{\perp}$ estão ativos, há uma sincronização e eles terminam. Quando $C^{\perp}$, $A^{\perp}$ é ativado, que por sua vez se comunica com A. Há então uma sincronização. Após essa sincronização, A termina, causando a ativação de B. Finalmente, ocorre a sincronização entre B e $B^{\perp}$.

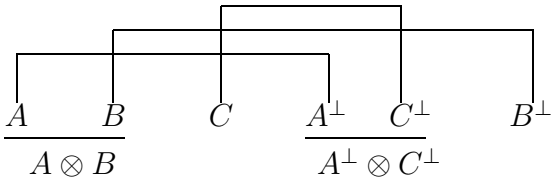

Figura 8: Sistema livre de 'deadlock'

4.8. Conclusão

Nessa breve introdução à lógica linear é importante ressaltar o estudo da chamada "geometria das deduções", proposto por Girard através do conceito de *redes-de-prova*, por apresentar uma perspectiva diferente de se estudar provas formais. Uma análise mais aprofundada dessa perspectiva aponta relações com um trabalho bastante anterior: a tese de doutorado de Statman [Sta74], na qual é proposto o estudo da complexidade estrutural de provas em dedução natural através de uma representação grafo-teórica das derivações. Um trabalho bastante recente, que também está inserido nessa perspectiva, refere-se a um modelo combinatorial, proposto por Carbone e Semmes [CaS00], para estudar o fenômeno da expansão de provas após a eliminação do corte no cálculo de seqüentes clássico [Car99].

5. Teoria de Tipos

Como diz D. van Dalen [vanD94], ao utilizar a chamada 'lógica clássica', ou seja, a lógica baseada em valores-verdade (verdadeiro ou falso), a tendência é de se guiar pela seguinte extrapolação (aparentemente inofensiva) da experiência com conjuntos finitos: universos infinitos podem ser examinados em sua totalidade. Em particular podemos ou não, de maneira global, determinar se a relação "estrutura A satisfaz a sentença $\exists x P(x)$" se verifica. Adaptando a frase de Hermann Weyl: estamos acostumados a pensar em conjuntos infinitos não apenas como sendo definidos por uma propriedade, mas como conjuntos cujos elementos são, por assim dizer, espalhados em nossa frente, de modo que podemos examinar um por um tal como um funcionário na delegacia examina seu arquivo. Segundo um matemático holandês do início do século chamado L. E. J. Brouwer (ver *Brouwer's Intuitionism* de W. van Stigt [vanS90]), essa visão do universo matemático é uma idealização atraente porém um tanto irrealista. Se se leva a sério nossas limitações perante as totalidades infinitas, então é preciso que se leia um enunciado como "existe um número primo maior que $10^{10^{10}}$" de maneira mais rígida que "é impossível que o conjunto dos primos se acabe antes de $10^{10^{10}}$". Pois não podemos inspecionar o conjunto dos números naturais de uma só vez e detectar um número primo. Temos que *exibir* um primo p maior que $10^{10^{10}}$.

Igualmente, pode-se estar convencido de que um certo problema (e.g. a determinação de um ponto de saturação de um jogo de soma-zero) tem uma solução na base de um

teorema abstrato (tal como o teorema do ponto fixo de Brouwer). Não obstante, pode-se sempre exibir uma solução. O que se precisa é de um método (prova) de *construção* que determine a solução.

Evidentemente, enunciados podem ser lidos de maneira não-construtiva. Na matemática a prática do raciocínio e dos procedimentos construtivos tem sido defendida por um número de pessoas, mas os fundadores da matemática construtiva são claramente L. Kronecker e L.E.J. Brouwer. Esse último apresentou um programa completo para a reconstrução da matemática numa base construtiva. A matemática de Brouwer (e a lógica que a acompanha) é chamada de *intuicionística*, e nesse contexto a matemática tradicional não-construtiva (e sua lógica) é chamada de *clássica*.

Há um número de questões filosóficas relacionadas com o intuicionismo, sobre as quais recomendamos ao leitor que consulte a literatura (cf. Dummett [Dum75], Troelstra & van Dalen [TrvanD88].

Argumentando que não se deve basear as interpretações da lógica na ficção de que o universo matemático é uma totalidade pré-determinada que pode ser examinada como um todo, A. Heyting teve que fornecer uma interpretação heurística dos conectivos lógicos na lógica intuicionística. Uma semântica semelhante foi proposta por A. Kolmogorov em 1932 [Kol32], desta vez a noção fundamental era a de 'problema': uma proposição da lógica construtiva na verdade definia um problema cuja solução seria atingida pela solução determinada pela estrutura da proposição; a interpretação via provas veio então a ser chamada de a interpretação de Brouwer–Heyting–Kolmogorov (BHK).

Parte-se do pressuposto de que um enunciado P é considerado verdadeiro (ou que se verifica) se se tem uma prova para ele. Por uma prova pretende-se dizer uma construção matemática que estabeleça P, e não uma dedução em um certo sistema formal. Por exemplo, uma prova de '$2 + 3 = 5$' consiste das sucessivas construções de $2, 3$ e 5, seguidas por uma construção que adiciona 2 a 3, seguida de uma construção que compara o resultado dessa adição e 5.

A noção primitiva aqui é "c prova P", onde entendemos por uma prova uma construção (para nossos propósitos, não especificada). Indicaremos agora como provas de enunciados compostos dependem de provas de suas partes.

($\wedge$) c prova $A \wedge B := c$ é um par $\langle a, b\rangle$ tal que a prova A e b prova B.
($\vee$) c prova $A \vee B := c$ é um par $\langle a, b\rangle$ tal que b é um número natural e se $a = 0$ então b prova A, e se $a \neq 0$ então b prova B.
($\rightarrow$) a prova $A \rightarrow B := c$ é uma construção que converte uma prova qualquer p de A em uma prova $c(p)$ de B.
(Λ) nenhum c prova Λ.

Para lidar com quantificadores assumimos que algum domínio D de objetos seja dado.

($\forall$) c prova $\forall x P(x) := c$ é uma construção tal que para cada $a \in D$, $c(a)$ prova $P(a)$.
($\exists$) c prova $\exists x P(x) := c$ é um par $\langle a, b\rangle$ tal que $a \in D$ e b prova $P(a)$.

Essa explanação dos conectivos serve como um meio de se dar ao leitor uma idéia sobre o que é e o que não é correto na lógica intuicionística. Geralmente considera-se a explanação como a definição do significado intuicionístico pretendido dos conectivos.

O conceito de 'realizabilidade' de Kleene. Uma das abordagens mais largamente aceitas da aritmética intuitionística que procura explicar esta última por meio da matemática clássica foi iniciada por S. Kleene no seu artigo seminal 'On the interpretation of intuitionistic number theory' [Kle45]. Tudo começa pela definição do predicado de realização 'e realiza P' dada por Kleene em 1945 e que usa uma recursão meta-matemática sobre a fórmula P da seguinte maneira:

Seja o predicado $e\circledR P$ (leia 'o número e realiza a fórmula P'), definido por indução estrutural sobre fórmulas como se segue:

$e\circledR A \wedge B$	é $\exists a \exists b[e = 2^a \cdot 3^b \wedge a\circledR A \wedge b\circledR B]$.
$e\circledR A \vee B$	é $\exists a[e = 2^0 \cdot 3^a \wedge a\circledR A] \vee \exists b[e = 2^1 \cdot 3^b \wedge b\circledR B]$.
$e\circledR A \rightarrow B$	é $\forall a[a\circledR A \rightarrow \exists y[T_1(e, a, y) \wedge U(y)\circledR B]]$.
$e\circledR \exists x A(x)$	é $\exists x \exists a[e = 2^x \cdot 3^a \wedge a\circledR A(x)]$.
$e\circledR \forall x A(x)$	é $\forall x \exists y[T_1(e, x, y) \wedge U(y)\circledR A(x)]$.

Portanto, o predicado de realizabilidade $\mathbb{P}$ de uma fórmula P é definido como sendo $\exists e[e\circledR P]$. Em outras palavras, o predicado de $\mathbb{P}$ se aplica a uma fórmula P se existe um número e tal que e realiza P.

Traduzindo a definição de Heyting para a notação do λ-cálculo. Tomando uma notação mais formal, a explicação heurística BHK dos conectivos lógicos ficaria da seguinte forma:

uma prova da proposição	tem a forma de:
$A \wedge B$	um par ordenado $\langle a, b\rangle$ onde a é uma prova de A e b é uma prova de B
$A \vee B$	um par ordenado $\langle a, 0\rangle$ onde a é uma prova de A ou $\langle b, 1\rangle$ onde b é uma prova de B
$A \rightarrow B$	é uma função $\lambda x.b(x)$ que associa uma prova a de A a uma prova $b(a)$ de B
$\forall x^D.P(x)$	é uma função $\lambda x.p(x)$ que associa um elemento qualquer d de D a uma prova $p(d)$ de $P(d)$
$\exists x^D.P(x)$	é um par ordenado $\langle d, p(d)\rangle$ onde d é um elemento do domínio D e $p(d)$ é uma prova de $P(d)$

Como o leitor pode notar, a explicação dos conectivos lógicos dada por Heyting estabelecem as condições sob as quais se tem uma prova da proposição correspondente. Sua contrapartida na Dedução Natural de Gentzen é a regra de *introdução*, agora enriquecida com a construção 'testemunha' que deverá ser manuseada pelo cálculo funcional.

Daí, as apresentações formais correspondentes *à la* Dedução Natural são, por exemplo:

$\wedge$-*introdução*

$$\frac{a : A \qquad b : A}{\langle a, b\rangle : A \wedge B}$$

$\vee$-*introdução*

$$\frac{a : A}{esq(a) : A \vee B} \qquad \frac{b : B}{dir(b) : A \vee B}$$

$\rightarrow$-*introdução*

$$\frac{\begin{array}{c}[x:A]\\ b(x):B\end{array}}{\lambda x.b(x):A\rightarrow B}$$

$\forall$-*introdução*

$$\frac{\begin{array}{c}[x:D]\\ p(x):P(x)\end{array}}{\lambda x.p(x):\forall x^{D}.P(x)}$$

$\exists$-*introdução*

$$\frac{a:D \qquad p(a):P(a)}{\langle a,p(a)\rangle:\exists x^{D}.P(x)}$$

Os operadores que constroem as provas canônicas são chamados 'construtores' (e.g. 'λ', '$\langle\ ,\ \rangle$', 'esq/dir'), enquanto que os operadores de eliminação que forma as provas não-canônicas são chamados 'destrutores'. (Obs.: 'esq' é um operador que marca a prova a como sendo uma prova vinda da proposição à esquerda, e 'dir' o operador à direita respectivo.)

5.1. A teoria de funcionalidade de Curry

A definição dos 'blocos básicos da lógica matemática', como Schönfinkel [Sch24] os chamou, foi dada originalmente por meio de regras de 'conversão'. Tomando *aplicação* como a noção mais primitiva, cada combinador foi então caracterizado pela maneira em que sua aplicação a um lista ordenada de objetos combinatórios operaria naquela lista, que objeto resultante seria obtido. Assim, escrevendo-se '$\rightarrow$' para abreviar 'converte para', variáveis ($x, y, z, ...$) para denotar objetos combinatórios arbitrários, os combinadores seriam definidos como:

$$\begin{array}{rl}\mathsf{I}x \rightarrow & x\\ \mathsf{B}xyz \rightarrow & x(yz)\\ \mathsf{B}'xyz \rightarrow & y(xz)\\ \mathsf{C}xyz \rightarrow & (xz)y\\ \mathsf{W}xy \rightarrow & (xy)y\\ \mathsf{S}xyz \rightarrow & (xz)(yz)\\ \mathsf{K}xy \rightarrow & x\end{array}$$

Esta foi essencialmente a caracterização original dos combinadores dada (independentemente) por Schönfinkel e Curry no final dos anos vinte. Mais tarde, em seu artigo de 1934 [Curry34], entretanto, Curry quis caracterizar os combinadores caracterizados por suas 'funcionalidades' (i.e. dados objetos de determinados tipos, o combinador retornaria um objeto de um certo tipo). Para fazer isso, Curry começou por atribuir 'tipos' aos respectivos objetos combinatórios. Obviamente, o princípio de aplicação permaneceria válido, apenas extendidos com símbolos para os tipos dos objetos, i.e.:

$$\text{(AP)} \qquad \frac{y:A \qquad x:A\rightarrow B}{ap(x,y):B}$$

significando: se y é do tipo A e x é do tipo $A \rightarrow B$ então a aplicação de x a y resulta em um objeto do tipo B. (A notação '$ap(x, y)$' torna explícito o operador de aplicação 'ap' de um objeto x em um objeto y.)

Assim, ao chegar numa conclusão que um objeto '$ap(x, y)$' é do tipo 'B' poderíamos seguir regra (AP) em direção contrária (da conclusão às premissas) e atribuir a x um tipo da forma '$A \to B$' para um certo A, e a y o tipo A propriamente dito. Seguindo tal procedimento poder-se-ia chegar na atribuição correta do tipo dos combinadores.

De modo a ver o procedimento funcionando, tomemos um dos combinadores, digamos 'B', por exemplo, e procuremos sua atribuição (mínima, ou principal) de tipo. Inicialmente, temos que:

$$\mathsf{B}xyz \to ap(x, ap(y, z))$$

A partir desse ponto começamos a atribuir ao objeto resultante, ou seja '$ap(x, ap(y, z))$', um tipo, digamos B. Daí,

$$ap(x, ap(y, z)) : \quad \mathsf{B}$$

Dito isto, sigamos (AP) no sentido inverso: enquanto o tipo de x deve ser da forma $A \to B$ para um certo A, $ap(y, z)$ deve ter este mesmo A como seu tipo. Por isso, nesse estágio teremos:

$$\begin{array}{rl} x : & A \to B \\ ap(y, z) : & A \end{array}$$

Da última linha deduzimos que enquanto y deve ser do tipo $C \to A$ para um certo C, z deve ter o mesmo C como seu tipo. Nesse ponto identificamos o tipo de nossos objetos combinatórios arbitrários:

$$\begin{array}{rl} x : & A \to B \\ y : & C \to A \\ z : & C \end{array}$$

Agora, precisamos encontrar o tipo do combinador B propriamente dito, que toma x, y, z (nessa ordem) e produz $ap(x, ap(y, z))$ do tipo B. Para fazer isso, faremos o movimento inverso do tipo do objeto resultante em direção ao iníicio da lista ordenada dos objetos de entrada, introduzindo um '$\to$' sempre que movemos uma posição na lista. Daí, teremos:

$$ap(x, ap(y, z)) : \quad B$$

construimos os seguintes passos:

$$\begin{array}{rl} \mathsf{B}xyz : & B \\ \mathsf{B}xy : & (C \to B) \\ \mathsf{B}x : & ((C \to A) \to (C \to B)) \\ \mathsf{B} : & (A \to B) \to ((C \to A) \to (C \to B)) \end{array}$$

e obtemos a funcionalidade (tipagem) de 'B' como

$$(A \to B) \to ((C \to A) \to (C \to B))$$

Se repetirmos o mesmo procedimento para os outros combinadores encontramos os seguintes esquemas de tipo:

$$\begin{array}{rl} \mathsf{I} : & A \to A \\ \mathsf{C} : & (A \to (B \to C)) \to (B \to (A \to C)) \\ \mathsf{W} : & (A \to (A \to B)) \to (A \to B) \\ \mathsf{S} : & (A \to (B \to C)) \to ((A \to B) \to (A \to C)) \\ \mathsf{K} : & A \to (B \to A) \end{array}$$

que podem ser lidos como os esquemas de axioma da lógica implicacional se '$\to$' for lido como 'implica'.

5.2. A contribuição de Howard

Em 1969, num manuscrito circulado entre os especialistas da área, que acabou sendo publicado em 1980, W. Howard [How80] estendeu a idéia da funcionalidade de Curry para todos os outros conectivos lógicos intuicionísticos. Nesse mesmo manuscrito a tal 'interpretação funcional' da lógica intuicionística ganhou a denominação de 'noção de construção *fórmulas-como-tipos*' (em inglês, '*formulae-as-types* notion of construction'). Da introdução do artigo:

> "H. Curry (1958) observou que existe uma correspondência íntima entre *axiomas* da lógica proposicional implicacional positiva, por um lado, e *combinadores básicos* por outro lado. Por exemplo, o combinador $\mathsf{K} = \lambda x.\lambda y.x$ corresponde ao axioma $\alpha \supset (\beta \sup \alpha)$.
> A noção de construção que se segue, para a lógica proposicional implicacional positiva, foi motivada pela observação de Curry. Mais precisamente, a observação de Curry forneceu *metade* da motivação. A outra metade foi fornecida pela descoberta de W. Tait da correspondência íntima entre eliminação do corte e redução de λ-termos (W. W. Tait, 1965).

Após adicionar os operadores sobre construções de prova para a negação, a conjunção, a disjunção, e a quantificação universal, Howard demonstra os teoremas de existência da forma normal para cada um dos conectivos lógicos, chegando ao teorema que diz que "todo termo pode ser reduzido a uma forma irredutível". Dado que o quantificador existencial apresenta-se como o menos fácil de formalização via o mesmo paradigma de 'fórmulas-como-tipos', Howard conclui o artigo com a definição das regras de construção para $\exists y.\alpha(y)$ em duas versões: 'existência fraca' e 'existência forte', enunciando e demonstrando o teorema da forma normal para tal operador lógico.

5.3. A teoria de tipos de Martin-Löf

Seguindo a idéia de Howard, e com o propósito de formalizar a matemática construtiva, Martin-Löf publicou em 1971 a primeira versão do que ficou conhecido como *Teoria Intuicionística de Tipos* [Mar71]. Devido à inclusão de uma regra sobre universos de tipos (i.e. 'tipos de tipos') que estabelecia a existência de um tipo universal contendo todos os tipos, e a conseqüente demonstração (por J.-Y. Girard, publicada em 1972) de sua inconsistência, a versão de 1971 acabou reformulada e em 1975 surgiu uma versão mais definitiva da 'parte predicativa' da teoria intuicionística de tipos [Mar75]. (As versões anteriores, incluindo a de 1972 [Mar72], continham regras 'impredicativas' pois incluíam regras para manuseio com universos de tipos entre as quais havia o enunciado da existência de um 'tipo universal', i.e. que continha todos os tipos inclusive a si próprio.)

Num artigo bastante influente publicado nos anais do *Sexto Congresso de Lógica, Metodologia e Filosofia da Ciência* [Mar82] (realizado em Hannover em Agosto de 1979) Martin-Löf associa definitivamente as construções da matemática construtiva com as instruções utilizadas em linguagens de programação, despertando daí um enorme interesse da comunidade de ciência da computação na teoria de tipos. Do resumo do artigo, "se programação for entendida não como a escrita de instruções para essa ou aquela máquina de computação mas como o desenho de métodos de computação que é a tarefa do computador de executar (uma diferença entre ciência da computação e ciência de computação), então não parece mais ser possível distinguir a disciplina de programação

da matemática construtiva". Mais adiante ele justifica porque a teoria intuicionística de tipos, que foi originalmente desenvolvida como um simbolismo para codificação precisa da matemática construtiva, pode também ser vista como uma linguagem de programação. Mais ainda, "as regras de inferência da teoria de tipos, que são completamente formais, aparecem como regras de síntese de programas corretos. Daí a corretude de um programa escrito na teoria de tipos é demonstrada formalmente ao mesmo tempo que ela está sendo sintetizada."

Com esse espírito, Martin-Löf traça um paralelo entre noções de programação e suas contrapartidas na matemática construtiva:

programação	matemática
programa, procedimento, algoritmo	função
entrada	argumento
saída, resultado	valor
$x := e$	$x = e$
$S_1; S_2$	composição de funções
se B **então** S_1 **senão** S_2	definição por casos
enquanto B **faça** S	definição por recursão
estrutura de dados	elemento, objeto
tipo de dados	conjunto, tipo
valor de um tipo de dados	elemento de um conjunto, objeto de um tipo
$a : A$	$a \in A$
inteiro	$\mathbb{Z}$
real	$\mathbb{R}$
Booleano	$\{0, 1\}$
$(c_1, \ldots, c_n)$	$\{c_1, \ldots, c_n\}$
array$[I]$ **de** T	T^I, $I \to T$
record $s_1 : T_1$; $s_2 : T_2$ **end**	$T_1 \times T_2$
record case $s : (c_1, c_2)$ **of** $c_1 : (s_1 : T_1)$; $c_2 : (s_2 : T_2)$ **end**	$T_1 + T_2$
conjunto de T	$\{0, 1\}^T$, $T \to \{0, 1\}$

De modo a concretizar a afirmação de que "as regras de inferência da teoria de tipos, que são completamente formais, aparecem como regras de síntese de programas corretos", e que portanto "a corretude de um programa escrito na teoria de tipos é demonstrada formalmente ao mesmo tempo que ela está sendo sintetizada", Martin-Löf define as chamadas "formas primitivas de expressão usadas na teoria de tipos", que na verdade são provenientes da associação de fórmulas a tipos. Senão vejamos:

forma canônica	forma não-canônica	forma lógica
$(\Pi x \in A)B$ $(\lambda x)b$	$c(a)$	$A \to B$ e $\forall x^A.B(x)$
$(\Sigma x \in A)B$, $\langle a, b\rangle$	$(Ex, y)(c, d)$	$A \wedge B$ e $\exists x^A.B(x)$
$A + B$ $esq(a)$, $dir(b)$	$(Dx, y)(c, d, e)$	$A \vee B$
$I(A, a, b)$ r	$J(c, d)$	$= (a, b)$
$\mathbb{N}_0$	$R_0(c)$	
$\mathbb{N}_1$ 0_1	$R_1(c, c_0))$	
$\mathbb{N}_2$ 0_2, 1_2	$R_2(c, c_0, c_1))$	
$\vdots$		
$\mathbb{N}$, 0, a'	$(Rx, y)(c, d, e)$	
$(Wx \in A)B$, $sup(a, b)$	$(Tx, y, z)(c, d)$	
U_0, U_1, $\ldots$		

Os operadores R e T corresponderiam a operadores de recursão primitiva e transcendente, respectivamente. Para maiores detalhes, o leitor deve se remeter ao artigo original.

Ao utilizar o tipo $(\Pi x \in A)B$ para corresponder às duas fórmulas lógicas $A \to B$ e $\forall x^A.B(x)$ Martin-Löf se vale do fato de que a interpretação BHK acima mencionada faz uso, para ambos os conectivos, da noção de "função que transforma uma prova (ou um objeto) em uma prova". Por outro lado, ao utilizar o tipo $(\Pi x \in A)B$ com a intenção de corresponder às duas fórmulas lógicas $A \wedge B$ e $\exists x^A.B(x)$ Martin-Löf também se vale da interpretação BHK (pois em ambos os casos se faz uso da noção de "par ordenado"). Embora formalmente aceitável, do ponto de vista lógico, no entanto, essa junção de '$\exists$' com '$\wedge$' causa uma certa confusão conceitual, pois o quantificador existencial tem afinidade lógica com a disjunção, e não com a conjunção. Pode-se pensar o enunciado $\exists x^A.B(x)$ como sendo $B(a_1) \vee B(a_2) \vee \ldots \vee B(a_n)$ se $a_1, \ldots, a_n$ for uma enumeração do domínio A. De qualquer forma, seguindo a definição formal dada por Martin-Löf, o operador pode ser particularizado tanto para as operações de projeção sobre um par ordenado, como para o operador de eliminação do quantificador existencial.

Uma versão posterior à de 1979 acabou sendo a mais utilizada pela comunidade científica de ciência da computação: trata-se da monografia intitulada *Intuitionistic Type Theory*, publicada em 1984 pela Editora Bibliopolis (Nápoles, Itália) baseada em uma série de notas de aula produzidas por G. Sambin a partir de uma seqüência de palestras proferidas em 1980 na Universidade de Pádua (Itália). Nessa versão algumas das construções são modificadas, notadamente aquela que se refere ao tipo 'igualdade' $I(A, a, b)$. Na verdade, a associação de um tipo com a proposição que enuncia a igualdade entre termos tem sido problemática desde as versões iniciais da teoria de tipos, criando uma 'dicotomia' artificial entre 'versão intencional' (1975) e 'versão extensional' (1984), tendo a versão de 1979 se colocado numa posição intermediária. Em [dQG94] dissolvemos tal dicotomia e apresentamos uma solução via 'dedução natural rotulada'.

5.4. A dedução natural rotulada

O uso de lógica na resolução de problemas de ciência da computação e de áreas afins, tal como a modelagem dos processos cognitivos através da chamada inteligência artificial, tem crescido muito. Obviamente, as necessidades da ciência da computação são diferentes daquelas que até então estimulavam o desenvolvimento da lógica como ciência. As motivações essencialmente relacionadas à formalização e à prova de consistência da matemática, além daquelas concernentes ao estudo da filosofia foram até então as principais fontes de problemas/desafios para os lógicos. Com o avanço da ciência da computação, uma decorrência dessa 'mudança' das origens dos problemas a serem resolvidos pelo uso de lógica foi exatamente um certo desvio no foco de atenção: determinados aspectos do conceito de 'lógica', que antes não tinham merecido tanta atenção, passaram a ser olhados com mais cuidado devido ao constante 'chamado' da ciência da computação. Dois desses aspectos, na opinião de D. Gabbay, têm importância crucial para as necessidades atuais da ciência da computação:

(i) As características de 'meta'-nível dos sistemas lógicos.
(ii) A 'lógica' das funções de Skolem e da unificação.

De modo geral, as características de meta-nível de sistemas lógicos encontram-se 'escondidas' na linguagem objeto, seja através da teoria de prova, ou mesmo via dispositivos de alta ordem. Ao mesmo tempo, não existe um tratamento sistemático das funções de dependência (as funções de Skolem). A tradição do uso de métodos de dedução baseados em

formas normais tem dominado, e uma das decorrências disso é o pouco desenvolvimento de técnicas de Skolemização em tempo de execução (em inglês, *run-time Skolemisation*). Um tratamento sistemático da 'lógica' das funções de Skolem será de grande utilidade na mudança desse estado de coisas.

No que concerne ao aspecto (i), observa-se claramente, no contexto de lógica aplicada à computação, uma proliferação 'desenfreada' de lógicas. Se, por um lado, novas aplicações 'pedem' a definição de lógica apropriada, por outro lado, as características diferenciadoras de cada uma dessas novas lógicas são, em sua maioria, provenientes de considerações de 'meta'-nível. Um dos aspectos menos positivos dessa proliferação de lógicas no contexto da ciência da computação, pode-se dizer, é exatamente: a complicação de formalismos: com o intuito de 'refletir' os aspectos menos declarativos na própria linguagem das fórmulas, é comum se observar a incorporação de elementos imperativos diretamente na lógica, o que leva à introdução de complicações no formalismo, e consequentemente, no seu entendimento e manuseio.

Desidérios

Seria conveniente:

1. dispor de uma perspectiva unificadora (o cálculo de seqüentes por si só não resolverá, como veremos adiante) que permita a 'fatoração' (em inglês, *factoring out*) das características de meta-nível daquelas essencialmente de nível objeto;
2. manter as regras de inferência tão simples quanto possível, ao mesmo tempo em que dispondo de um cálculo que manuseie as características de meta-nível *em separado* e de forma tal que mantenha uma certa *harmonia* com o cálculo lógico sobre as fórmulas;
3. dispor de meios para a estruturação e a combinação de lógicas;
4. fazer com que as suposições relevantes em uma dedução sejam apropriadamente 'trazidas à superfície': isso contribuiria para que se pudesse ter meios de se tratar a explicitação e o uso de recursos, tópicos bastante evidenciados com a definição da lógica linear por J.-Y. Girard [Gir87a].

Por que razão o cálculo de seqüentes por si só não resolverá

O cálculo de seqüentes, tendo sido concebido como um 'meta'-cálculo de provas tal que suprisse a falta de simetria do cálculo de dedução natural, presta-se bastante às dualidades e simetrias da lógica clássica. Em $\Gamma \vdash \Delta$, que deve ser lido como 'a conjunção das proposições em Γ implica na disjunção das proposições em Δ', a dualidade Booleana aparece naturalmente: um lado conjuntivo, outro disjuntivo; um lado 'negativo', outro 'positivo'; as regras de introdução à esquerda têm uma forma semelhante às regras de introdução à direita do conectivo dual respectivo; etc.

Entretanto, desde Frege, lógica é *também* sobre quantificadores, predicados, funções, igualdade, objetos, referentes, etc. Se, por um lado, o càlculo de seqüentes se apresenta como um 'meta'-cálculo de provas, embora esse cálculo seja 'meta' somente enquanto lida com a 'história' da dedução unicamente baseada nas fórmulas usadas nela, um cálculo rotulado se prestará melhor à condição de 'meta'-cálculo, na medida em que

puder manusear diretamente a 'história' da dedução baseada também nos referentes e funções utilizados no curso da dedução.

Em resumo: além das dualidades da lógica Booleana, a lógica Fregeana também lida com *quantificação* de forma *direta* (ao invés de usar, digamos, diagramas de Venn), por isso uma análise apropriada da dedução em um cálculo com quantificadores deve permitir a análise e o manuseio direto de *dependências* entre referentes. Evidentemente, essa não é uma característica do cálculo de seqüentes.

O que seriam os rótulos?

A informação adicionada ao longo das fórmulas têm o papel, como já vimos, de trazer algumas das características de meta-nível de volta para o nível objeto de forma disciplinada, e tal que não complique demasiadamente a linguagem das fórmulas. O rótulo, com sua origem num domínio essencialmente procedural, constitui informação de natureza bem diferente daquela contida nas fórmulas (estas últimas constituindo-se essencialmente de declarações): é como se o rótulo fosse o portador das informações do tipo *o que posso mostrar* (i.e. referentes, funções, etc.), enquanto que as fórmulas estariam sendo portadoras d*o que posso dizer*.

De modo geral, podemos então fazer uso de várias leituras para a unidade declarativa '$t : A$' dos sistemas dedutivos rotulados:

1. Um valor de confiabilidade difuso (um número x, $0 \leq x \leq 1$). Essa leitura pode ser útil em sistemas especialistas que usam lógicas difusas.
2. A origem da fórmula (de onde vem a informação). Pode ser útil em sistemas de gerenciamento de bancos de dados complexos.
3. A prioridade da fórmula. Útil na modelagem de sistemas dedutivos com prioridade.
4. A construção de prova da fórmula. Aqui a interpretação funcional de Curry–Howard se encarrega do resto.

O leitor que prefere permanecer com a perspectiva mais tradicional de que lógica é a respeito de:

suposições (dados) provando conclusões

poderia ver as fórmulas rotuladas como um tipo especial de informação declarativa (i.e. enriquecida com informação adicional).

Em muitas ocasiões a apresentação de um elemento de informação da forma $t : P$, onde t é o rótulo e P a fórmula, é bastante intuitiva. Em geral, o uso de t como rótulo significa que t representa uma certa informação necessária à modificação e/ou suplementação da informação que P representa, mas que não é do mesmo tipo da informação representada por P propriamente.

Regras da dedução natural rotulada

As regras são definidas em blocos: *introdução*, *redução* (igualdade do tipo β) e *indução* (igualdade do tipo η):

$\wedge$-*introdução*

$$\frac{a_1 : A_1 \qquad a_2 : A_2}{\langle a_1, a_2 \rangle : A_1 \wedge A_2}$$

$\vee$-*introdução*

$$\frac{a_1 : A_1}{esq(a_1) : A_1 \vee A_2} \qquad \frac{a_2 : A_2}{dir(a_2) : A_1 \vee A_2}$$

$\rightarrow$-*introdução*

$$\frac{\begin{array}{c}[x : A] \\ b(x) : B\end{array}}{\lambda x.b(x) : A \rightarrow B}$$

$\forall$-*introdução*

$$\frac{\begin{array}{c}[x : D] \\ f(x) : P(x)\end{array}}{\lambda x.f(x) : \forall x^D.P(x)}$$

$\exists$-*introdução*

$$\frac{a : D \qquad f(a) : P(a)}{\varepsilon x.(f(x), a) : \exists x^D.P(x)}$$

β-redução

$\wedge$-*redução*

$$\frac{\dfrac{a_1 : A_1 \qquad a_2 : A_2}{\langle a_1, a_2 \rangle : A_1 \wedge A_2}\ \wedge\textit{-intr}}{prim(\langle a_1, a_2 \rangle) : A_1}\ \wedge\textit{-elim} \quad \twoheadrightarrow_\beta \quad a_1 : A_1$$

$$\frac{\dfrac{a_1 : A_1 \qquad a_2 : A_2}{\langle a_1, a_2 \rangle : A_1 \wedge A_2}\ \wedge\textit{-intr}}{seg(\langle a_1, a_2 \rangle) : A_2}\ \wedge\textit{-elim} \quad \twoheadrightarrow_\beta \quad a_2 : A_2$$

$\vee$-*redução*

$$\frac{\dfrac{a_1 : A_1}{esq(a_1) : A_1 \vee A_2}\ \vee\textit{-intr} \quad \begin{array}{c}[s_1 : A_1] \quad [s_2 : A_2] \\ d(s_1) : C \quad e(s_2) : C\end{array}}{caso(esq(a_1), \upsilon s_1.d(s_1), \upsilon s_2.e(s_2)) : C}\ \vee\textit{-elim} \quad \twoheadrightarrow_\beta \quad \begin{array}{c}[a_1 : A_1] \\ d(a_1/s_1) : C\end{array}$$

$$\frac{\dfrac{a_2 : A_2}{dir(a_2) : A_1 \vee A_2}\ \vee\textit{-intr} \quad \begin{array}{c}[s_1 : A_1] \quad [s_2 : A_2] \\ d(s_1) : C \quad e(s_2) : C\end{array}}{caso(dir(a_2), \upsilon s_1.d(s_1), \upsilon s_2.e(s_2)) : C}\ \vee\textit{-elim} \quad \twoheadrightarrow_\beta \quad \begin{array}{c}[a_2 : A_2] \\ e(a_2/s_2) : C\end{array}$$

$\rightarrow$-*redução*

$$\frac{a : A \qquad \dfrac{\begin{array}{c}[x : A] \\ b(x) : B\end{array}}{\lambda x.b(x) : A \rightarrow B}\ \rightarrow\textit{-intr}}{ap(\lambda x.b(x), a) : B}\ \rightarrow\textit{-elim} \quad \twoheadrightarrow_\beta \quad \begin{array}{c}[a : A] \\ b(a/x) : B\end{array}$$

$\forall$-*redução*

$$\dfrac{a : D \qquad \dfrac{\begin{array}{c}[x : D]\\ f(x) : P(x)\end{array}}{\lambda x.f(x) : \forall x^D.P(x)}\forall\text{-}intr}{extr(\lambda x.f(x), a) : P(a)}\forall\text{-}elim \qquad \begin{array}{c}[a : D]\\ f(a/x) : P(a)\end{array}$$

$\exists$-*redução*

$$\dfrac{\dfrac{a : D \quad f(a) : P(a)}{\varepsilon x.(f(x), a) : \exists x^D.P(x)}\exists\text{-}intr \qquad \begin{array}{c}[t : D, g(t) : P(t)]\\ d(g, t) : C\end{array}}{inst(\varepsilon x.(f(x), a), \sigma g.\sigma t.d(g, t)) : C}\exists\text{-}elim \quad \twoheadrightarrow_\beta$$

$$\begin{array}{c}[a : D, f(a) : P(a)]\\ d(f/g, a/t) : C\end{array}$$

β-igualdade

$\wedge$-β-*igualdade*

$$prim(\langle a_1, a_2\rangle) =_\beta a_1 \qquad\qquad seg(\langle a_1, a_2\rangle) =_\beta a_2$$

$\vee$-β-*igualdade*

$$\begin{array}{c}caso(esq(a_1), \upsilon s_1.d(s_1), \upsilon s_2.e(s_2)) =_\beta d(a_1/s_1)\\ caso(dir(a_2), \upsilon s_1.d(s_1), \upsilon s_2.e(s_2)) =_\beta e(a_2/s_2)\end{array}$$

$\rightarrow$-β-*igualdade*

$$ap(\lambda x.b(x), a) =_\beta b(a/x)$$

$\forall$-β-*igualdade*

$$extr(\lambda x.f(x), a) =_\beta f(a/x)$$

$\exists$-β-*igualdade*

$$inst(\varepsilon x.(f(x), a), \sigma g.\sigma t.d(g, t)) =_\beta d(f/g, a/t)$$

η-redução

$\wedge$-*indução*

$$\dfrac{\dfrac{c : A_1 \wedge A_2}{prim(c) : A_1}\wedge\text{-}elim \qquad \dfrac{c : A_1 \wedge A_2}{seg(c) : A_2}\wedge\text{-}elim}{\langle prim(c), seg(c)\rangle : A_1 \wedge A_2}\wedge\text{-}intr \qquad \twoheadrightarrow_\eta \qquad c : A_1 \wedge A_2$$

$\vee$-*indução*

$$\dfrac{c : A_1 \vee A_2 \quad \dfrac{[a_1 : A_1]}{esq(a_1) : A_1 \vee A_2}\vee\text{-}intr \quad \dfrac{[a_2 : A_2]}{dir(a_2) : A_1 \vee A_2}\vee\text{-}intr}{caso(c, \upsilon a_1.esq(a_1), \upsilon a_2.dir(a_2)) : A_1 \vee A_2}\vee\text{-}elim \ \twoheadrightarrow_\eta$$

$$c : A_1 \vee A_2$$

$\rightarrow$-*indução*

$$\dfrac{\dfrac{[x : A] \qquad c : A \rightarrow B}{ap(c, x) : B}\rightarrow\textit{-elim}}{\lambda x.ap(c, x) : A \rightarrow B}\rightarrow\textit{-intr} \qquad \twoheadrightarrow_\eta \qquad c : A \rightarrow B$$

onde c não depende de x.

$\forall$-*indução*

$$\dfrac{\dfrac{[t : D] \qquad c : \forall x^D.P(x)}{extr(c, t) : P(t)}\forall\textit{-elim}}{\lambda t.extr(c, t) : \forall t^D.P(t)}\forall\textit{-intr} \qquad \twoheadrightarrow_\eta \qquad c : \forall x^D.P(x)$$

onde x não ocorre livre em c.

$\exists$-*indução*

$$\dfrac{c : \exists x^D.P(x) \quad \dfrac{[t : D] \qquad [g(t) : P(t)]}{\varepsilon y.(g(y), t) : \exists y^D.P(y)}\exists\textit{-intr}}{inst(c, \sigma g.\sigma t.\varepsilon y.(g(y), t)) : \exists y^D.P(y)}\exists\textit{-elim} \quad \twoheadrightarrow_\eta \quad c : \exists x^D.P(x)$$

η-igualdade

$\wedge$-η-*igualdade*

$$\langle prim(c), seg(c)\rangle =_\eta c$$

$\vee$-η-*igualdade*

$$caso(c, \upsilon a_1.\texttt{esq}(a_1), \upsilon a_2.\texttt{dir}(a_2)) =_\eta c$$

$\rightarrow$-η-*igualdade*

$$\lambda x.ap(c, x) =_\eta c$$

desde que x não ocorra livre em c.

$\forall$-η-*igualdade*

$$\lambda t.extr(c, t) =_\eta c$$

desde que c não tenha ocorrências livres de x.

$\exists$-η-*igualdade*

$$inst(c, \sigma g.\sigma t.\varepsilon y.(g(y), t)) =_\eta c$$

ζ-reduções

$\vee$-ζ-*redução*

$$\dfrac{\dfrac{p : A_1 \vee A_2 \quad \begin{matrix}[s_1 : A_1] \\ d(s_1) : C\end{matrix} \quad \begin{matrix}[s_2 : A_2] \\ e(s_2) : C\end{matrix}}{caso(p, \upsilon s_1.d(s_1), \upsilon s_2.e(s_2)) : C}}{w(caso(p, \upsilon s_1.d(s_1), \upsilon s_2.e(s_2))) : W} \twoheadrightarrow_\zeta$$

$$\dfrac{p : A_1 \vee A_2 \quad \dfrac{[s_1 : A_1]}{\dfrac{d(s_1) : C}{w(d(s_1)) : W}} \quad \dfrac{[s_2 : A_2]}{\dfrac{e(s_2) : C}{w(e(s_2)) : W}}}{caso(p, \upsilon s_1.w(d(s_1)), \upsilon s_2.w(e(s_2))) : W}$$

$\exists$-ζ-*redução*

$$\dfrac{\dfrac{e : \exists x^D.P(x) \quad \dfrac{[t : D, g(t) : P(t)]}{d(g,t) : C}}{inst(e, \sigma g.\sigma t.d(g,t)) : C}}{w(inst(e, \sigma g.\sigma t.d(g,t))) : W} \quad \twoheadrightarrow_\zeta$$

$$\dfrac{e : \exists x^D.P(x) \quad \dfrac{[t : D, g(t) : P(t)]}{\dfrac{d(g,t) : C}{w(d(g,t)) : W}}}{inst(e, \sigma g.\sigma t.w(d(g,t))) : W}$$

ζ-igualdade

$$w(caso(p, \upsilon s_1.d(s_1), \upsilon s_2.e(s_2)), u) =_\zeta caso(p, \upsilon s_1.w(d(s_1)), \upsilon s_2.w(d(s_2)))$$

$$w(inst(e, \sigma g.\sigma t.d(g,t)), u) =_\zeta inst(e, \sigma g.\sigma t.w(d(g,t), u))$$

5.4.1. Exemplo de construção de uma prova

Como vimos anteriormente, na dedução natural rotulada temos uma regra de *introdução* e uma de *eliminação* para cada conectivo lógico, mais um cálculo de *rotulação* ao lado das fórmulas, fazendo o papel do dispositivo de meta-nível. (Pode-se pensar no sinal de derivação do cálculo de seqüentes como tal dispositivo no cálculo logístico de Gentzen.)

Vamos ver como as regras funcionam. De modo a verificar se a fórmula

$$(C \to (A \to B)) \to ((C \to A)) \to (C \to B))$$

é válida (i.e. pode ser provada sem deixar qualquer suposição em aberto) analisamos a fórmula em suas subpartes, e começamos a construir as subdeduções:

Para a fórmula ser válida devemos começar de:

$$x : C \to (A \to B) \tag{1}$$

como uma suposição, e chegar em

$$f(x) : (C \to A) \to (C \to B) \tag{2}$$

(para alguma expressão $f(x)$) como premissa de uma regra de $\to$-*introdução*.

Agora, analisando (2) sabemos que a expressão '$f(x)$' deve ser um λ-termo, dado que o conectivo principal é uma implicação. Portanto, temos

$$\lambda y.g(x,y) : (C \to A) \to (C \to B) \tag{3}$$

(para alguma expressão g que depende de ambos x e y) que deve ter vindo de uma $\to$-*introdução*, com

$$y : C \to A \tag{4}$$

como suposição, e

$$g(x,y) : C \to B \tag{5}$$

como premissa. Agora, (5) tem 'implicação' como seu conectivo principal, portanto, (5) deve ser da forma:

$$\lambda w.h(x,y,w) : C \to B \tag{6}$$

e deve ter vindo de

$$w : C \tag{7}$$

como uma suposição, e

$$h(x,y,w) : B \tag{8}$$

como premissa da $\to$-*introdução*.

Agora temos:
(1) $x : C \to (A \to B)$
(4) $y : C \to A$
(7) $w : C$

como suposições, e precisamos obter B a partir delas, o que 'resolverá' nossa desconhecida '$h(x,y,w)$' (h é uma expressão que depende de x, y e w). Agora, de (1) e (7) aplicando $\to$-*eliminação* obtemos

$$ap(x,w) : (A \to B) \tag{9}$$

De (4) e (7) por $\to$-*eliminação* obtemos

$$ap(y,w) : A \tag{10}$$

Agora, de (9) e (10) por $\to$-*eliminação* obtemos

$$ap(ap(x,w), ap(y,w)) : B \tag{11}$$

e agora temos o rótulo de B como uma expressão dependendo de x, y and w (i.e. nossa desconhecida h).

Procedendo com nossa dedução da fórmula inteira a partir das subdeduções, por $\to$-*introdução* obtemos

$$\lambda w.ap(ap(x,w), ap(y,w)) : C \to B \tag{12}$$

Por $\to$-*introdução* obtemos

$$\lambda y.\lambda w.ap(ap(x,w), ap(y,w)) : (C \to A) \to (C \to B) \tag{13}$$

Novamente, por $\to$-*introdução* obtemos

$$\lambda x.\lambda y.\lambda w.ap(ap(x,w), ap(y,w)) : (C \to (A \to B)) \to ((C \to A \to (C \to B)) \tag{14}$$

e terminamos: todas as suposições introduzidas no processo de trabalho com as subdeduções foram descartadas, como as variáveis-rótulo correspondentes revelam (i.e. elas estão todas ligadas por alguma abstração). A árvore de prova pode agora ser construída.

Referências

S. Abramsky. Proof as processes. *TCS*, 135:05-09, 1994.

Peter H. G. Aczel. Frege Structures and the Notions of Proposition, Truth and Set. In J. Barwise, H.-J. Keisler, and K. Kunen, editors, *The Kleene Symposium*, vol. 101 of *Studies in Logic and The Foundations of Mathematics*, pages 31–59, Amsterdam, xx+425pp, 1980. North-Holland Publishing Co. Proc. of the Symposium held in June 18–24, 1978, at Madison, Wisconsin, USA.

Yuuki Andou. A Normalization-Procedure for the First Order Classical Natural Deduction with Full Logical Symbols. *Tsukuba Journal of Mathematics*, 19(1):153-162, 1995.

A. Asperti. Causal Dependencies in Multiplicative Linear Logic with MIX. *Math. Struct. in Comp. Sci.*, 5:351-380, 1995.

A. Asperti and G. Dore. Yet another soundness criterion for multiplicative Linear Logic with MIX. *Logical Foundations of Comp. Sci.*, San Petersburgh. 1994.

A. Carbone. Duplication of directed graphs and exponential blow-up of proofs. *APAL*, 100:1-76, 1999.

A. Carbone and S. Semmes. *A Graphic Apology for Symmetry and Implicitness*. Oxford Mathematical Monographs. OUP, 2000.

Edson Holanda Cavalcante Jr. *Sobre os Critérios de Corretude para os Grafos de Prova* Dissertação de mestrado, Centro de Informática, Univ. Fed. de Pernambuco, C.P. 7851, Recife-PE, 2001.

N. C. da Costa. *Introdução aos Fundamentos da Matemática*. HUCITEC, São Paulo, 1952.

H. B. Curry. Functionality in Combinatory Logic. *Proc. of the National Academy of Sciences of USA* 20: 584–590, 1934

V. Danos and L. Regnier. The structure of multiplicatives. *Archive for Math. Logic*, 28:181–203,1989.

M.Davis, R. Sigal and E.J.Weyuker Computability, Complexity, and Languages: Fundamentals of Theoretical Computer Science In Werner Rheinboldt, editor, *Computer Science and Scientific Computing: A Series of Monographs and Textbooks*. Academic Press. San Diego, 1994.

Anjolina Grisi de Oliveira. *Proofs from a Geometric Perspective* PhD thesis, Centro de Informática, Univ. Fed. de Pernambuco, C.P. 7851, Recife-PE, 2001.

Ruy J. G. B. de Queiroz. *Proof Theory and Computer Programming. The Logical Foundations of Computation*. PhD thesis, Department of Computing, Imperial College, University of London, February 1990.

Ruy J. G. B. de Queiroz. *Grundgesetze* alongside *Begriffsschrift* (abstract). In *Abstracts of Fifteenth International Wittgenstein Symposium*, pages 15–16, 1992. Symposium held in Kirchberg/Wechsel, August 16–23 1992.

Dosen, K. Sequent-Systems for Modal Logics. *JSL*, 30(1):149–68,1985.

Ruy J. G. B. de Queiroz and Dov M. Gabbay. An introduction to labelled natural deduction. In *Third Advanced Summer School in Artificial Intelligence*, 1992. To appear in the *Bulletin of the IGPL*. Proceedings of the Summer School held at S. Miguel, Azores, September 21–25 1992.

Ruy J. G. B. de Queiroz and Dov M. Gabbay. Equality in Labelled Deductive Systems and the functional interpretation of propositional equality. In P. Dekker and M. Stokhof, editors, *Proceedings of the 9th Amsterdam Colloquium*, pages 547–546, 1994.

Michael A. E. Dummett. The Philosophical Basis of Intuitionistic Logic. In H. E. Rose and J. C. Shepherdson, editors, *Logic Colloquium '73*, vol. 80 of *Studies in Logic and The Foundations of Mathematics*, pages 5–40, Amsterdam, viii+513pp, 1975. North-Holland. Proceedings of the Colloquium held in Bristol, UK, 1973.

Frege, Gottlob (1879). *Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*. Verlag von Louis Nebert, Halle. English translation 'Begriffsschrift, *a formula language, modeled upon that of arithmetic, for pure thought*' in [vanH67], pages 1–82.

Frege, Gottlob (1893). *Grundgesetze der Arithmetik. Begriffsschriftlich abgeleitet. I*. Verlag von Hermann Pohle, Jena. Reprinted in vol. 32 of *Olms Paperbacks*, Georg Olms Verlagsbuchhandlung, Hildesheim, 1966, XXXII+254pp. Partial English translation in [Fur64].

Gottlob Frege. *Grundgesetze der Arithmetik. Begriffsschriftlich abgeleitet. II*. Verlag von Hermann Pohle, Jena, 1903. Reprinted in vol. 32 of *Olms Paperbacks*, Georg Olms Verlagsbuchhandlung, Hildesheim, 1966, XVI+266pp. Partial English translation in [Gea52].

Montgomery Furth, editor. *The Basic Laws of Arithmetic. Exposition of the System*. University of California Press, Berkeley and Los Angeles, lxiv+143pp, 1964. Partial English translation of Gottlob Frege's *Grundgesetze der Arithmetik*.

Dov M. Gabbay. *Labelled Deductive Systems, Vol. I - Foundations.* OUP, 1996.

Dov M. Gabbay and Ruy J. G. B. de Queiroz. Extending the Curry-Howard intepretation to linear, relevant and other resource logics. *JSL* 57(4):1319–1365.

Peter Geach and Max Black, editors. *Translations from the Philosophical Writings of Gottlob Frege*. Basil Blackwell, Oxford, x+228pp, 3rd (1980) edition, 1952.

Gerhard Gentzen. Untersuchungen über das logische Schliessen. *Mathematische Zeitschirift*, pages 176–210 and 405–431, 1935. English translation:"Investigations into Logical Deduction" in *The Collected Works of Gehard Gentzen*, ed. M.E.Szabo, North-Holland Pub Co.,1969.

Girard, Jean-Yves (1971). Une Extension de l'Interpretation de Gödel à l'Analyse, et son Application à l'Elimination des Coupures dans l'Analyse et la Théorie des Types. In *Proceedings of the Second Scandinavian Logic Symposium* vol. 63 of *Studies in Logic and The Foundations of Mathematics*, pages 63–92. North-Holland, Amsterdam, viii+405pp. Proceedings of the Symposium held in Oslo, June 18–20 1970.

Jean-Yves Girard. Linear logic. *TCS*, 50:1–102, 1987.

Jean-Yves Girard. *Proof Theory and Logical Complexity*. Vol. 1 of *Studies in Proof Theory*. Bibliopolis, Naples, 1987.

Jean-Yves Girard. Towards a geometry of interaction. In *Categories in Computer science and Logic*, vol. 92 of *Contemporary Mathematics*, pages 69-108. AMS Publications, 1989.

J. Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*. Cambridge University Press, 1989.

Gödel, Kurt (1958). Über eine bisher noch nicht benützte Erweiterung des finiten Standpunktes. *Dialectica* 12:280–287. English translation 'On a hitherto unexploited extension of the finitary standpoint' in *Journal of Philosophical Logic*, 9:133–142, 1980.

Nicolas D. Goodman. A theory of constructions equivalent to arithmetic. In *Intuitionism and Proof Theory*. Proceedings of the *Summer Conference* at Buffalo, New York, 1968, A. Kino, J. Myhill & R. E. Vesley (eds.). Série Studies in Logic and The Foundations of Mathematics, North-Holland, vii+516pp, pp. 101–120.

I. Hacking. What is Logic?. In *Journal of Philosophy*, vol. LXXVI, n.6, 1979. pp. 285-319.

Heyting, Arend. Die formale Regeln der intuitionistische Logik. *Sitzungsberichte der preuszischen Akademie von Wissenschaften (physicalischmathematische Klasse)* 42–56, 1930.

Arend Heyting. *Intuitionism. An Introduction*. Series *Studies in Logic and the Foundations of Mathematics*. North-Holland, Amsterdam, viii+133pp, 1956.

W. A. Howard. The formulae-as-types notion of construction. In J. R. Seldin and J.R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic Lambda Calculus and Formalism*. Academic Press, 1980. xxv+606pp. Privately circulated notes, 1969, only later published in Curry's *Festschrift*.

S. C. Kleene. On the Interpretation of Intuitionistic Number Theory. *JSL*, 10:109–124, 1945.

S. C. Kleene. *Introduction to Metamathematics*. Van Nostrand, Princeton, 1952.

A. N. Kolmogorov. Zur deutung der intuitionistischen logik. *Mathematische Zeitschrift*, 35:58–65, 1932.

Lambek, Joachim and Scott, P. J. *Introduction to higher order categorical logic*, vol. 7 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge, ix+293pp.

D. Leivant Assumptions classes in natural deduction. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 25:1-4, 1979.

Per Martin-Löf. A Theory of Types. Report 71-3, Department of Mathematics, University of Stockholm, 1971. 57pp. February 1971, revised October 1971.

Per Martin-Löf. An intuitionistic theory of types. Report, Mathematical Institute, University of Stockholm, 1972. 86pp.

Per Martin-Löf. An intuitionistic theory of types: predicative part. In H. E. Rose and J. C. Shepherdson, editors, *Logic Colloquium '73*, vol. 80 of *Studies in Logic and The Foundations of Mathematics*, pages 73–118, Amsterdam, viii+513pp, 1975. North-Holland. Proceedings of the Colloquium held in Bristol, UK, in 1973.

Per Martin-Löf. Constructive Mathematics and Computer Programming. In L. J. Cohen, J. Łos, H. Pfeiffer, and K.-P. Podewski, editors, *Logic, Methodology and Philosophy of Science VI*, Series *Studies in Logic and The Foundations of Mathematics*, pages 153–175, Amsterdam, xiii+738pp, 1982. North-Holland. Proceedings of the International Congress held in Hannover, August 22–29 1979.

Cosme D. B. Massi *Provas de Normalização para a Lógica Clássica*. PhD thesis, Dep. de Filosofia do Instituto de Filosofia e Ciências Humanas da Universidade Estadual de Campinas - SP, 1990.

Masini, A. 2-Sequent Calculus: Intuitionism and Natural Deduction. *JLC*, 3(5):533–62, 1993.

A. T. Martins. *A Syntactical and Semantical Uniform Treatment for the IDL & LEI Nonmonotonic System.* PhD thesis, Dep. de Informática, Univ. Federal de Pernambuco, Recife, 1997. Supervisor: T. Pequeno.

A. T. Martins and T. Pequeno. A Sequent Calculus for the Logic of Epistemic Inconsistency. In T. Pequeno e F. Carvalho, editors, *Proc. of the 11th Brazilian Symposium on Artificial Intelligence*. Fortaleza, Brazil, Oct. 17-20, 1994.

A. T. Martins, T. Pequeno, and M. Pequeno. A Sequent Calculus for a Paraconsistent Default Logic. In R. de Queiroz and W. Carnielli, editors, *Proc. of the 6th Workshop on Logic, Language, Information and Computation*. Itatiaia, May, 25-28, 1999. pp. 139-149.

Luiz Carlos Pereira and Cosme Massi. Normalização para a lógica clássica. *O que nos faz pensar*, 2:49-53, 1990.

Jan von Plato. Proof Theory of full classical propositional logic. Tech. Rep. Dep. of Philosophy, Univ. of Helsinki, 1998.

D. Prawitz. *Natural Deduction. A Proof-Theoretical Study*, vol. 3 of *Acta Universitatis Stockholmiensis. Stockholm Studies in Philosophy*. Almqvist & Wiksell, Stockholm, 113pp, 1965.

D. Prawitz. Ideas and results in proof theory. In J. E. Fenstad, editor, *Proceedings of the Second Scandinaviam Logic Symposium*. North-Holland, 1971.

D. Prawitz. Validity and normalizability of proofs in first and second order classicam and intuitionistic logic. In *Atti del Congresso Nazionale di Logica*, pp. 11-36, Montecatini, 1979.

N. Prior. The runabout Inference-ticket. *Analysis*, 21:2, 1960.

Moses Schönfinkel. Über die bausteine der mathematischen logik. *Mathematische Annalen*, 92:305–316, 1924. English translation 'On the building blocks of mathematical logic' in [vanH67], pages 355–366.

Dana S. Scott. Constructive Validity. In Proceedings of the *Symposium on Automated Deduction*, held in Versailles, Dezembro, 1968. Editado por M. Laudet, D. Lacombe, L. Nolin & M. Schützenberger, série Lecture Notes in Mathematics, vol. 125, Springer-Verlag, v+310pp, pp. 237–275.

Jonathan Seldin. Normalization and Excluded Middle I. *Studia Logic* XLVIII, pp. 193-217, 1989.

Gunnar Stalmarck. Normalization Theorems for Full First Order Classical Natural Deduction. *JSL*, 56(1):129-149, March 1991.

R. Statman *Structural Complexity of Proofs*. PhD thesis, Stanford University, 1974.

William W. Tait. Infinitely long terms of transfinite type. In J. N. Crossley and M. A. E. Dummett, editors, *Formal Systems and Recursive Functions*, Series *Studies in Logic and The Foundations of Mathematics*, pages 176–185, Amsterdam, 320pp, 1965. North-Holland. Proceedings of the *Logic Colloquium '63*, held in Oxford, UK.

William W. Tait. Intensional interpretations of functionals of finite type I. *JSL*, 32:198–212, 1967.

A. S. Troelstra and D. van Dalen. *Constructivism in Mathematics: An Introduction. Vol. II*, vol. 123 of *Studies in Logic and The Foundations of Mathematics*. North-Holland, Amsterdam, xvii+535pp, 1988.

D. van Dalen. *Logic and Structure*, 3rd edition, Springer-Verlag.

Jean van Heijenoort, editor. *From Frege to Gödel: A Source Book in Mathematical Logic. 1879-1931*. Series *Source Books in the History of the Sciences*. Harvard University Press, Cambridge, Massachussetts, xii+664pp, 1967.

Walter P. van Stigt. *Brouwer's Intuitionism*, vol. 2 of *Studies in the History and Philosophy of Mathematics*. North-Holland, Amsterdam, XXVI+530pp, 1990.

A.S.Troelstra and H.Schwichtenberg. *Proof Theory* Cambridge University Press, 1996.

A. M. Ungar. *Normalization, Cut-elimination and the Theory of Proofs*. Number 28 of CSLI Lecture Notes, Center for the Study of Language and Information, Stanford, 1992.