1 Cálculo Lambda não-tipado ($\lambda_{\beta\eta}$)

A teoria dos tipos possui como história de origem algumas tentativas falhas. O conceito de tipos pode ser mapeado para dois matemáticos importantes que fizeram usos bem diferentes dele: Bertrand Russel (e Walfred North Whitehead) na Principia Mathematica e Alonzo Church no seu Cálculo λ simplesmente tipado (STλC).

A teoria dos tipos que é usada hoje, provém do segundo autor e de outros autores que vêm dessa tradição. Por isso, o início dessas notas se propõe a começar do básico, definindo o que é o Cálculo λ não tipado e quais questões levaram Church a desenvolver a teoria dos tipos em cima dele.

Aqui, será traduzido "λ-calculus" como "Cálculo λ", decisão que perde a estética do hífen, mas que mantém a unidade com outras traduções de "X calculus" no corpo matemático brasileiro, como o "Cálculo Diferencial e Integral", o "Cálculo de sequentes", o "Cálculo de variações", etc.

1.1 O Cálculo

1.1.1 Definições

O cálculo lambda serve como uma abstração em cima do conceito de função. Uma função é uma estrutura que pega um *input* e retorna um *output*, por exemplo a função $f(x) = x^2$ pega um input x e retorna seu valor ao quadrado x^2. No cálculo lambda, essa função pode ser denotada por $\lambda x.x^2$, onde $\lambda x.$ simboliza que essa função espera receber como entrada x. Quando se quer saber qual valor a função retorna para uma entrada específica, são usados números no lugar das variáveis, como por exemplo $f(3) = 3^2 = 9$. No cálculo lambda, isso é feito na forma de $(\lambda x.x^2)(3)$.

Esses dois principrios de construção são definidos como:

- **Abstração**: Seja M uma expressão e x uma variável, podemos construir uma nova expressão $\lambda x.M$. Essa expressão é chamada de Abstração de x sobre M
- **Aplicação**: Sejam M e N duas es expressões, podemos construir uma expressão MN. Essa expressão é chamada de Aplicação de M em N.

Dadas essas operações, é preciso também de uma definição que dê conta do processo de encontrar o resultado após a aplicação em uma função. Esse processo é chamado de β-redução. Ela faz uso da substituição e usa como notação os colchetes.

Definição 1.1 (β-redução)**.** A β-redução é o processo de resscrita de uma expressão da forma $(\lambda x.M)N$ em outra expressão $M[x := N]$, ou seja, a expressão M na qual todo x foi substituido por N.

1.1.2 Sintáxe do Cálculo Lambda

É interessante definir a sintaxe do cálculo lambda de forma mais formal. Para isso, são utilizados métodos que podem ser familiares para aqueles que já trabalharam com lógica proposicional, lógica de primeira ordem ou teoria de modelos.

Primeiro, precisamos definir a linguagem do Cálculo λ.

Definição 1.2. (i) Os *termos lambda* são palavras em cima do seguinte alfabeto:

- variáveis: $v_0, v_1, \ldots$
- abstrator: λ
- parentesis: $(,)$

(ii) O conjunto de λ-termos Λ é definido de forma indutiva da seguinte forma:

- Se x é uma variável, então $x \in \Lambda$
- $M \in \Lambda \rightarrow (\lambda x.M) \in \Lambda$
- $M, N \in \Lambda \rightarrow MN \in \Lambda$

Na teoria dos tipos e no cálculo lambda, é utilizada uma forma concisa de definir esses termos chamada de Formalismo de Backus-Naur ou Forma Normal de Backus (BNF, em inglês). Nessa forma, a definição anterior é reduzida à:

$$\Lambda = V|(\Lambda\Lambda)|(\lambda V\Lambda)$$

Onde V é o conjunto de variáveis $V = \{x, y, z, \ldots\}$

Para expressar igualdade entre dois termos de Λ utilizamos o simbolo $\equiv$.

Algumas definições indutivas podem ser formadas a partir da definição dos λ-termos.

Definição 1.3 (Multiconjunto de subtermos)**.**

1. (Base) $Sub(x) = \{x\}$, para todo $x \in V$
2. (Aplicação) $Sub((MN)) = Sub(M) \cup Sub(N) \cup \{(MN)\}$
3. (Abstração) $Sub((\lambda x.M)) = Sub(M) \cup \{(\lambda x.M)\}$

Observações:
(i) Um subtermo pode ocorrer múltiplas vezes, por isso é escolhido chamar de multiconjunto (ii) A abstração de vários termos ao mesmo tempo pode ser escrita como $\lambda x.(\lambda y.x)$ ou como $\lambda xy.x$.

Lema 1.1 (Propriedades de Sub)**.**

- (Reflexividade) Para todo λ-termo M, temos que $M \in Sub(M)$
- (Transitividade)Se $L \in Sub(M)$ e $M \in Sub(N)$, então $L \in Sub(N)$.

Definição 1.4 (Subtermo próprio)**.** L é um subtermo próprio de M se L é subtermo de M e $L \not\equiv M$

Exemplos:

1. Seja o termo $\lambda x.\lambda y.xy$, vamos calcular seus subtermos:

$$\begin{aligned} Sub(\lambda x.\lambda y.xy) &= \{\lambda x.\lambda y.xy\} \cup Sub(\lambda y.xy) \\ &= \{\lambda x.\lambda y.xy\} \cup \{\lambda y.xy\} \cup Sub(xy) \\ &= \{\lambda x.\lambda y.xy\} \cup \{\lambda y.xy\} \cup Sub(x) \cup Sub(y) \\ &= \{\lambda x.\lambda y.xy, \lambda y.xy, x, y\} \end{aligned}$$

2. Seja o termo $(y(\lambda x.(xyz)))$, vamos calcular os seus subtermos:

$$\begin{aligned} Sub(y(\lambda x.(xyz))) &= Sub(y) \cup Sub((\lambda x.(xyz))) \\ &= \{y\} \cup \{(\lambda x.(xyz))\} \cup Sub((xyz)) \\ &= \{y\} \cup \{(\lambda x.(xyz))\} \cup Sub(x) \cup Sub(y) \cup Sub(z) \\ &= \{y\} \cup \{(\lambda x.(xyz))\} \cup \{x\} \cup \{y\} \cup \{z\} = \{y, (\lambda x.(xyz)), x, y, z\} \end{aligned}$$

Outro conjunto importante para a sintaxe do cálculo lambda é o de variáveis livres. Uma variável é dita *ligante* se está do lado do λ. Em um termo $\lambda x.M$, x é uma variável ligante e toda aparição de x em M é chamada de *ligada*. Se existir uma variável em M que não é ligante, então dizemos que ela é *livre*. Por exemplo, em $\lambda x.xy$, o primeiro x é ligante, o segundo x é ligado e y é livre.

O conjunto de todas as variáveis livres em um termo é denotado por FV e definido da seguinte forma:

Definição 1.5 (Multiconjunto de variáveis livres)**.**

1. (Base) $FV(x) = \{x\}$, para todo $x \in V$
2. (Aplicação) $FV((MN)) = FV(M) \cup FV(N) \cup \{(MN)\}$
3. (Abstração) $FV((\lambda x.M)) = FV(M) \setminus \{x\}$

Exemplos:

1. Seja o termo $\lambda x.\lambda y.xyz$, vamos calcular seus subtermos:

$$\begin{aligned} FV(\lambda x.\lambda y.xyz) &= FV(\lambda y.xyz) \setminus \{x\} \\ &= FV(xyz) \setminus \{y\} \setminus \{x\} \\ &= FV(x) \cup FV(y) \cup FV(z) \setminus \{y\} \setminus \{x\} \\ &= \{x, y, z\} \setminus \{y\} \setminus \{x\} \\ &= \{z\} \end{aligned}$$

Vamos definir os termos fechados da seguinte forma:

Definição 1.6. O λ-termo M é dito *fechado* se $FV(M) = \emptyset$. Um λ-termo fechado também é chamado de *combinador*. O conjunto de todos os λ-termos fechados é chamado de Λ^0.

Os combinadores são muito utilizados na *Lógica Combinatória*, mas vamos explorá-los mais a frente.

1.1.3 Conversão

No cálculo Lambda, é possível renomear variáveis ligantes/ligadas, pois a mudança dos nomes dessas variáveis não muda a sua interpretação. Por exemplo, $\lambda x.x^2$ e $\lambda u.u^2$ podem ser utilizadas de forma igual, mesmo que com nomes diferentes. A Renomeação será definida da seguinte forma:

Definição 1.7. Seja $M^{x \to y}$ o resultado da troca de todas as livre-ocorrencias de x em M por y. A relação de renomeação é expressa pelo símbolo $=_\alpha$ e é definida como: $\lambda x.M =_\alpha \lambda y.M^{x \to y}$, dado que $y \notin FV(M)$ e y não seja ligante em M.

Podemos extender essa definição para a definição do renomeamento, chamado de α-conversão.

Definição 1.8 (α-conversão)**.**

1. (Renomeamento) $\lambda x.M =_\alpha \lambda y.M^{x \to y}$
2. (Compatibilidade) Sejam M, N, L termos. Se $M =_\alpha N$, então $ML =_\alpha NL$, $LM =_\alpha LN$
3. (Regra ξ) Para um z qualquer, se $M =_\alpha N$, então $\lambda z.M =_\alpha \lambda z.N$
4. (Reflexividade) $M =_\alpha M$
5. (Simetria) Se $M =_\alpha N$, então $N =_\alpha M$
6. (Transitividade) Se $L =_\alpha M$ e $M =_\alpha N$, então $L =_\alpha N$

A partir dos pontos (4), (5) e (6) dessa definição, é possível dizer que a α-conversão é uma relação de equivalência, chamada de α-equivalência.

Exemplos:

1. $(\lambda x.x(\lambda z.xy)) =_\alpha (\lambda u.u(\lambda z.uy))$
2. $(\lambda x.xy) \neq_\alpha (\lambda y.yy)$

1.1.4 Substituição

Podemos definir agora a substituição de um termo por outro da seguinte forma:

Definição 1.9 (Substituição)**.**

1. $x[x := N] \equiv N$
2. $y[y := x] \equiv y$, se $x \not\equiv y$
3. $(PQ)[x := N] \equiv (P[x := N])(Q[x := N])$
4. $(\lambda y.P)[x := N] \equiv (\lambda z.P^{y \to z})[x := N]$ se $(\lambda z.P^{y \to z})$ é α-equivalente a $(\lambda y.P)$ e $z \notin FV(N)$

A notação $[x := N]$ é uma meta-notação, pois não está definida na sintaxe do cálculo lambda. Na literatura também é possível ver a notação $[N/x]$ para definir a substituição.

1.1.5 Beta redução

Voltando à aplicação, agora com a substituição em mente, podemos dizer que a aplicação de um termo N em $\lambda x.M$, na forma de $(\lambda x.M)N$ é a mesma coisa que $M[x := N]$. Nesse caso, essa única substituição entre termos pode ser descrita na seguinte definição:

Definição 1.10 (β-redução para único passo)**.**

1. (Base) $(\lambda x.M)N \to_\beta M[x := N]$

2. (Compatibilidade) Se $M \to_\beta N$, então $ML \to_\beta NL$, $LM \to_\beta LN$ e $\lambda x.M \to_\beta \lambda x.N$

O termo $(\lambda x.M)N$ é chamado de *redex*, vindo do ingles "reducible expression" (expressão reduzível), e o subtermo $M[x := N]$ é chamado de *contractum* do redex.

Exemplos:

1. $(\lambda x.x(xy))N \to_\beta N(Ny)$
2. $(\lambda x.xx)(\lambda x.xx) \to_\beta (\lambda x.xx)(\lambda x.xx)$
3. $(\lambda x.(\lambda y.yx)z)v \to_\beta (\lambda y.yv)z \to_\beta zv$

Os exemplos 2 e 3 são importantes por duas razões:

- Com o exemplo 3 é possível ver que é possível concatenar várias reduções seguindas, vamos colocar uma definição mais geral a diante que lide com isso.
- Com o exemplo 2 é possível ver que existem termos que, quando beta-reduzidos, retornam eles mesmos. Isso faz com que cálculo lámbda não tipado tenha propriedades interessantes, pois muitas vezes a simplificação não termina. Ou seja, é possível haver cadeias de beta redução que não possuem termo mais simples.

Definição 1.11 (β-redução para zero ou mais passos)**.** $M \twoheadrightarrow_\beta N$ (lê-se: M beta reduz para N em vários passos) se existe um $n \geq 0$ e existem termos M_0 até M_n tais que $M_0 \equiv M$, $M_n \equiv N$ e para todo i tal que $0 \leq i < n$:

$$M_i \to_\beta M_{i+1}$$

Ou Seja:

$$M \equiv M_0 \to_\beta M_1 \to_\beta \cdots \to_\beta M_{n-1} \to_\beta M_n \equiv N$$

Lema 1.2.

1. $\twoheadrightarrow_\beta$ é uma extensão de $\to_\beta$, ou seja: se $M \to_\beta N$, então $M \twoheadrightarrow_\beta N$
2. $\twoheadrightarrow_\beta$ é reflexivo e transitivo, ou seja:
   - (reflexividade) Para todo M, $M \twoheadrightarrow_\beta M$
   - (transitividade) Para todo L, M, e N. Se $L \twoheadrightarrow_\beta M$ e $M \twoheadrightarrow_\beta N$, então $L \twoheadrightarrow_\beta N$

Prova

1. Na definição 1.11, seja $n = 1$, então $M \equiv M_0 \to_\beta M_1 \equiv N$, que é a mesma coisa que $M \to_\beta N$
2. Se $n = 0$, $M \equiv M_0 \equiv N$
3. A transitividade também segue da definição

Uma extensão dessa β-redução geral é a β- conversão, definida como:

Definição 1.12 (β-conversão)**.** $M =_\beta N$ (lê-se: M e N são β-convertíveis) se existe um $n \geq 0$ e existem termos M_0 até M_n tais que $M_0 \equiv M$, $M_n \equiv N$ e para todo i tal que $0 \leq i < n$: Ou $M_i \to_\beta M_{i+1}$ ou $M_{i+1} \to_\beta M_i$

Lema 1.3.

1. $=_\beta$ é uma extensão de $\twoheadrightarrow_\beta$ em ambas as direções, ou seja: se $M \twoheadrightarrow_\beta N$ ou $N \twoheadrightarrow_\beta M$, então $M =_\beta N$
2. $=_\beta$ é uma relação de equivalência, ou seja, possui reflexividade, simetria e transitividade
   - (reflexividade) Para todo M, $M =_\beta M$
   - (Simetria) Para todo M e N, se $M =_\beta N$, então $N =_\beta M$
   - (transitividade) Para todo L, M, e N. Se $L =_\beta M$ e $M =_\beta N$, então $L =_\beta N$

1.1.6 Forma Normal

Podemos definir a hora de parar de reduzir, para isso vamos introduzir o conceito de forma Normal

Definição 1.13 (Forma normal β ou β-normalização)**.**

1. M está na forma normal β se M não possui nenhum redex
2. M possui uma formal normal β, ou é β-normalizável, se existe um N na forma normal β tal que $M =_\beta N$. N é chamado de a *forma normal* β de M.

Lema 1.4. Se M está em sua forma normal β, então $M \twoheadrightarrow_\beta N$ implica em $M \equiv N$

Exemplos:

1. $(\lambda x.(\lambda y.yx)z)v$ tem como formal normal β zv, pois $(\lambda x.(\lambda y.yx)z)v \to_\beta zv$ (como visto nos exemplos anteriores) e zv está na forma normal β
2. Vamos definir um termo $\Omega := (\lambda x.xx)(\lambda x.xx)$, Ω não está na forma normal β, pois pode ser β-reduzido, mas não possui também forma normal β, pois ele sempre é β-reduzido para ele mesmo.
3. Seja $\Delta := (\lambda x.xxx)$, então $\Delta\Delta \to_\beta \Delta\Delta\Delta \to_\beta \Delta\Delta\Delta\Delta \to_\beta \ldots$. Logo $\Delta\Delta$ não possui forma normal.

Definição 1.14 (Caminho de Redução)**.**
Um caminho de redução finito de M é uma sequência finita de termos $N_0, N_1, \ldots, N_n$ tais que $N_0 \equiv M$ e $N_i \to_\beta N_{i+1}$, para todo $0 \leq i < n$.
Um caminho de redução infinito de M é uma sequência infinita de termos $N_0, N_1, \ldots$ tais que $N_0 \equiv M$ e $N_i \to_\beta N_{i+1}$, para todo $i \in \mathbb{N}$

Considerando esses dois tipos de caminhos de redução, vamos definir dois tipos de normalização

Definição 1.15 (Normalização Fraca e Forte)**.**

1. M é *fracamente normalizável* se existe um N na forma normal β tal que $M \twoheadrightarrow_\beta N$

2. M é *fortemente normalizável* se não existem caminhos de requção infinitos começando de M.

Todo termo M que é fortemente normalizável é fracamente normalizável.

Os termos Ω e Δ não são nem fortemente normalizáveis, nem fracamente normalizáveis.

É possível relacionar a normalização fraca com a forma normal β usando a intuição que, se M reduz para ambos N_1 e N_2, então existe um termo N_3 que exista no caminho de redução de ambos N_1 e N_2.

Teorema 1.1 (Teorema de Church-Rosser ou Teorema da Confluência)**.** Suponha que para um λ-termo M, tanto $M \twoheadrightarrow_\beta N_1$ e $M \twoheadrightarrow_\beta N_2$. Então existe um λ-termo N_3 tal que $N_1 \twoheadrightarrow_\beta N_3$ e $N_2 \twoheadrightarrow_\beta N_3$

A prova desse teorema pode ser encontrada no Livro de Barendregt.

Uma consequência importante desse teorema é que o resultado do calculo feito em cima do termo não depende da ordem que esses cálculos são feitos. A escolha dos redexes não interfere no resultado final.

Corolário 1.1.
Suponha que $M =_\beta N$. Então existe um termo L tal que $M \twoheadrightarrow_\beta L$ e $N \twoheadrightarrow_\beta L$.

prova. Como $M =_\beta N$, então, pela definição, existe um $n \in \mathbb{N}$ tal que:

$$M \equiv M_0 \rightleftarrows_\beta M_1 \dots M_{n-1} \rightleftarrows_\beta M_n \equiv N$$

. Onde $M_i \rightleftarrows_\beta M_{i+1}$ significa que ou $M_i \to_\beta M_{i+1}$ ou $M_{i+1} \to_\beta M_i$. Vamos provar por indução em n:

1. Quando $n = 0$: $M \equiv N$. Então sendo $L \equiv M$, $M \twoheadrightarrow_\beta L$ e $N \twoheadrightarrow_\beta L$ (por zero passsos)

2. Quando $n = k > 0$, entao existe M_{k-1}. Logo temos que $M \equiv M_0 \rightleftarrows_\beta M_1 \dots M_{k-1} \rightleftarrows_\beta M_k \equiv N$. Por indução, existe um L' tal que $M_0 \twoheadrightarrow_\beta L'$ e $M_{k-1} \twoheadrightarrow_\beta L'$. Vamos dividir $M_{k-1} \rightleftarrows_\beta M_k$ em dois casos

   (a) Se $M_{k-1} \to_\beta M_k$, então como $M_{k-1} \to_\beta M_k$ e $M_{k-1} \twoheadrightarrow_\beta L'$, então, pelo Teorema de Church-Rosser, existe um L tal que $L' \twoheadrightarrow_\beta L$ e $M_k \twoheadrightarrow_\beta L$. Logo encontramos L.

   (b) Se $M_k \to_\beta M_{k-1}$, então como $M_0 \twoheadrightarrow_\beta L'$ e $M_k \twoheadrightarrow_\beta L'$, L' é o próprio L.

   □.

Lema 1.5.

1. Se M possui forma normal β N, então $M \twoheadrightarrow_\beta N$.

2. Um λ-termo tem no máximo uma forma normal β

Prova

1. Seja $M =_\beta N$, com N como formal normal β. Então, pelo corolário anterior, existe um L tal que $M \twoheadrightarrow_\beta L$ e $N \twoheadrightarrow_\beta L$. Como N é a forma normal, N não é mais redutível e $N \equiv L$. Então $M \twoheadrightarrow_\beta L \equiv N$, logo $M \twoheadrightarrow_\beta N$.

2. Suponha que M possui duas formas normais β N_1 e N_2. Então por (1), $M \twoheadrightarrow_\beta N_1$ e $M \twoheadrightarrow_\beta N_2$. Pelo teorema de Church-Rosser, existe um L tal que $N_1 \twoheadrightarrow_\beta L$ e $N_2 \twoheadrightarrow_\beta L$. Mas como N_1 e N_2 estão na forma normal, $L \equiv N_1$ e $L \equiv N_2$. Então pela transitividade da equivalência, $N_1 \equiv N_2$.

   □.

1.1.7 Teorema do ponto fixo

No Cálculo λ, todo λ-termo L possui um *ponto fixo*, ou seja, existe um λ-termo M tal que $LM =_\beta M$. O termo Ponto Fixo vêm da análise funcional: seja f uma função, então f possui um ponto fixo a se $f(a) = a$.

Teorema 1.2. Para todo $L \in \Lambda$, existe um $M \in \Lambda$ tal que $LM =_\beta M$.

prova: Seja L um λ-termo e defina $M := (\lambda x.L(xx))(\lambda x.L(xx))$. M é um redex, logo:

$$\begin{aligned} M &\equiv (\lambda x.L(xx))(\lambda x.L(xx)) \\ &\to_\beta L((\lambda x.L(xx))(\lambda x.L(xx))) \\ &\equiv LM \end{aligned}$$

Logo $LM =_\beta M$. □

Pela prova anterior, podemos perceber que M pode ser generalizado para todo λ-termo. Esse M será denominado de *Combinador de ponto fixo* e escrito na forma:

$$Y \equiv \lambda y.(\lambda x.y(xx))(\lambda x.y(xx))$$

1.1.8 Eta redução

A junção da definição 0.8 com as definições de β-redução gera uma teoria que será chamada aqui de λ_β. Para essa teoria, faltam alguns detalhes que podem, ou não, ser introduzidos a depender do que se precisa.

A η-redução é a segunda redução possível dentro do cálculo λ. Através dela é possível remover uma abstração que não faz nada para o termo interior. Sua definição é:

Definição 1.16 (η-redução)**.**

1. $(\lambda x.Mx) \to_\eta M$, onde $x \notin FV(M)$.

A junção da teoria λ com a η-redução será chamada aqui de $\lambda_{\beta\eta}$.

Uma outra adição possível à teoria λ é chamada de extencionalidade e definida da seguinte forma:

Definição 1.17 (extencionalidade **Ext**)**.** Dados os termos M e N, se $Mx = Nx$ para todo λ-termo x, com $x \notin FV(MN)$, então $M = N$.

Ext introduz no cálculo λ a noção presente na teoria dos conjuntos de igualdade entre funções. Na teoria dos conjuntos, duas funções $f : A \to B$ e $g : A \to B$ são iguais se, para todo $x \in A$, $f(x) = g(x)$.

A união da teoria λ com **Ext** é chamada de $\lambda + \mathbf{Ext}$.

Teorema 1.3 (Teorema de Curry)**.** As teorias $\lambda_{\beta\eta}$ e $\lambda + \boldsymbol{Ext}$ são equivalentes.

Prova: Primeiro, é necessário mostrar que η é derivável de $\lambda + \boldsymbol{Ext}$. Seja a igualdade $(\lambda x.Mx)x = Mx$, por **Ext**, $\lambda x.Mx = M$.

Segundo, é necessário mostrar que dado $Mx = Nx$, é possível derivar $M = N$ em $\lambda_{\beta\eta}$. Para isso, seja $Mx = Nx$, realizando ξ-redução, tem-se que $\lambda x.Mx = \lambda x.Nx$. Fazendo η-redução dos dois lados, $M = N$. $\square$

Existe uma outra formulação da extencionalidade dentro do cálculo λ chamado de regra ω. É necessário um equivalente à **Ext** para restrições do cálculo λ que só possuem termos fechados, para isso, é desenvolvida a regra ω:

Definição 1.18 (Regra ω)**.** Dados os termos M e N, se $MQ = NQ$ para todo termo fechado Q, então $M = N$.

Da regra ω é possível deduzir **Ext**, mas não o oposto. A prova dessa dedução não será mostrada.

Posteriormente, será feito uma discussão de teorias dos tipos que aceitam **Ext** como um axioma no estilo de $\lambda + \boldsymbol{Ext}$ e outras que conseguem derivar a extencionalidade através de outras propriedades, como $\lambda_{\beta\eta}$.

1.1.9 Codificações dentro do Cálculo λ

O primeiro exemplo de transformação de funções em λ-termos, $f(x) = x^2$ para $\lambda x.x^2$, pode parecer correto, mas supõe mais que foi definido até então. Pois partindo somente da sintaxe e das transformações vistas nas seções anteriores, não foi definido coisas básicas como o que significa a exponenciação ou o número 2. Se o cálculo *lambda* é colocado como um possível substituto para a teoria das funções baseada na teoria dos conjuntos, então ele deve ser capaz de definir todas essas coisas de forma interna. Por isso, foram desenvolvidas as *codificações*, das quais a primeira e mais conhecida é a *Codificação de Church* (Church Encoding).

Primeiro, é necessário definir os números naturais e, para isso, é preciso de combinadores que traduzam os axiomas de Peano para os números naturais. Ou seja, precisamos definir o número 0 e a função sucessor $suc(x) = x + 1$. Para isso, diferente das outras definições indutivas vistas anteriormente, primeiro serão definidos os números e depois as operações.

Definição 1.19 (Numerais de Church)**.**

1. $zero := \lambda fx.x$
2. $um := \lambda fx.fx$
3. $dois := \lambda fx.f(fx)$
   ...
4. $n := \lambda fx.f^n x$

Onde $f^n x$ é $f(f(f \ldots x))$ n vezes.

As operações são descritas na forma:

Definição 1.20 (Operações aritméticas)**.**

1. $sum := \lambda m.\lambda n.\lambda fx.mf(nfx)$
2. $mult := \lambda m.\lambda n.\lambda fx.m(nf)x$
3. $suc := \lambda m.\lambda fx.f(mfx)$

Nessas definições os primeiros m e n são os números m e n, como por exemplo $m + n$, $m \times n$, $m + 1$, etc.

Exemplos:

1. Prova que $sum\ one\ one \twoheadrightarrow_\beta two$ na codificação:

$$\begin{aligned} sum\ one\ one &\equiv (\lambda m.\lambda n.\lambda fx.mf(nfx))\ one\ one \\ &\twoheadrightarrow_\beta (\lambda fx.onef(onefx)) \\ &\twoheadrightarrow_\beta (\lambda fx.(\lambda gx.gx)f((\lambda gx.gx)fx)) \\ &\twoheadrightarrow_\beta (\lambda fx.(\lambda x.fx)(fx)) \\ &\twoheadrightarrow_\beta (\lambda fx.f(fx)) \\ &\equiv two \end{aligned}$$

2. Prova que $mult\ two\ two \twoheadrightarrow_\beta four$ na codificação:

$$\begin{aligned} mult\ two\ two &\equiv (\lambda m.\lambda n.\lambda fx.m(nf)x)\ two\ two \\ &\twoheadrightarrow_\beta (\lambda fx.\ two(\ two\ f)x) \\ &\twoheadrightarrow_\beta (\lambda fx.(\lambda gy.g(gy))(\ two\ f)x) \end{aligned}$$

Uma vez definida a multiplicação e a soma, é possível definir outras operações como o fatorial e a exponenciação. Isso fica como exercício para o leitor.

Tendo definido operações relacionadas aos números naturais, pode-se perguntar se é possível construir algo lógico dentro do cálculo λ não-tipado. Para isso, é necessário definir a noção de "verdadeiro" e "falso", na forma:

Definição 1.21 (Booleanos)**.**

1. $true := \lambda xy.x$
2. $false := \lambda xy.y$
3. $not := \lambda z.z\ false\ true$
4. $'if\ x\ then\ u\ else\ v' := \lambda x.xuv$

Exemplos:

1. Prova que $not(not\ p) \equiv p$ na codificação:

$$\begin{aligned} not(not\ p) &\equiv not((\lambda z.z\ false\ true\ )p) \\ &\twoheadrightarrow_\beta not(p\ false\ true\ ) \\ &\twoheadrightarrow_\beta not(p(\lambda xy.y)(\lambda xy.x)) \\ &\twoheadrightarrow_\beta (\lambda z.z\ false\ true\ )(p(\lambda xy.y)(\lambda xy.x)) \\ &\twoheadrightarrow_\beta (p(\lambda xy.y)(\lambda xy.x))\ false\ true \end{aligned}$$

Se $p \twoheadrightarrow_\beta \mathit{true}$,

$$\begin{aligned} \mathit{not}(\mathit{not}\ \mathit{true}\ ) &\twoheadrightarrow_\beta ((\lambda xy.x)(\lambda xy.y)(\lambda xy.x))\ \mathit{false}\ \mathit{true} \\ &\twoheadrightarrow_\beta ((\lambda xy.y))\ \mathit{false}\ \mathit{true} \\ &\twoheadrightarrow_\beta \mathit{true} \end{aligned}$$

Se $p \twoheadrightarrow_\beta \mathit{false}$,

$$\begin{aligned} \mathit{not}(\mathit{not}\ \mathit{false}\ ) &\twoheadrightarrow_\beta ((\lambda xy.y)(\lambda xy.y)(\lambda xy.x))\ \mathit{false}\ \mathit{true} \\ &\twoheadrightarrow_\beta ((\lambda xy.x))\ \mathit{false}\ \mathit{true} \\ &\twoheadrightarrow_\beta \mathit{false} \end{aligned}$$

1.2 Modelos

Na matemática, um **modelo** é uma forma de dar sentido à estrutura sintática desenvolvida. No Cálculo λ, os primeiros modelos só foram desenvolvidos posteriormente à sintaxe, pois a simples descrição do cálculo na teoria dos conjuntos gerava inconsistências com os axiomas da teoria dos conjuntos.

1.2.1 Estruturas Aplicativas

Primeiro, antes de definir o que é um modelo, é necessário definir um tipo de estrutura algébrica:

Definição 1.22. Uma *Estrutura Aplicativa* é um par $\langle D, \bullet \rangle$, onde D é um conjunto com ao menos dois elementos, chamado de *domínio* da estrutura, e $\bullet$ é um mapeamento de $\bullet : D \times D \to D$.

Os modelos do Cálculo λ serão estruturas aplicativas acrecidas de propriedades extras. A condição de se ter pelo menos dois elementos em D é importante para evitar modelos triviais.

Seja $\mathcal{M} = \langle D, \bullet \rangle$ uma estrutura aplicativa, escreve-se $a \in \mathcal{M}$ caso $a \in D$.

Definição 1.23. Uma estrutura aplicativa $\mathcal{M} = \langle D, \bullet \rangle$ é *extensional* se para $a, b \in D$, têm-se que $\forall x \in D, a \bullet x = b \bullet x \Rightarrow a = b$. a e b são chamadas de *extensionalmente iguais* e são escritos como $a \sim b$.

Definição 1.24. Seja $\mathcal{M} = \langle D, \bullet \rangle$ uma estrutura aplicativa e seja $n \geq 1$. Uma função $\theta : D^n \to D$ é *representável* se, e somente se, D possui um membro a tal que:

$$(\forall d_1, \ldots, d_n \in D) a \bullet d_1 \bullet d_2 \bullet \cdots \bullet d_n = \theta(d_1, \ldots, d_n)$$

Usando a convenção de associação à esquerda, essa equação é lida como:

$$(\ldots((a \bullet d_1) \bullet d_2) \bullet \cdots \bullet d_n) = \theta(d_1, \ldots, d_n)$$

Cada a é chamado de *representante* de θ. O conjunto de todas as funções representáveis de D^n para D é chamado de $(D^n \to D)_{rep}$.

Definição 1.25. Uma *Algebra Combinatória* é uma estrutura aplicativa $\mathbb{D} = \langle D, \bullet \rangle$, onde dados $k, s \in D$,

1. $(\forall a, b \in D)\ k \bullet a \bullet b = a$
2. $(\forall a, b, c \in D)\ s \bullet a \bullet b \bullet c = a \bullet c \bullet (b \bullet c)$.

Uma Algebra combinatória também é chamada de uma estrutura *combinatorialmente completa*

1.2.2 Modelos interpretativos algébricos

O primeiro tipo de modelo para o Cálculo λ surge através das estruturas aplicativas da seguinte forma:

Definição 1.26. Um *modelo* de $\lambda\beta$ é uma tripla $\mathbb{D} = \langle D, \bullet, [\![\ ]\!] \rangle$, onde $\langle D, \bullet \rangle$ é uma estrutura aplicativa e $[\![\ ]\!]$ é um mapeamento que leva para cada λ-termo M e cada valuação ρ, um membro $[\![M]\!]_\rho$ de D tal que:

1. Para toda variável x, $[\![x]\!]_\rho = \rho(x)$
2. Para todos os termos M e N, $[\![MN]\!]_\rho = [\![M]\!]_\rho \bullet [\![N]\!]_\rho$
3. Para toda variável x, termo M e elemento $d \in D$, $[\![\lambda x.M]\!]_\rho \bullet d = [\![M]\!]_{[d/x]\rho}$
4. Para todo termo M e valuações ρ e σ, $[\![x]\!]_\rho = [\![x]\!]_\sigma$, toda vez que $\rho(x) = \sigma(x)$ para todas as variáveis livres x de M
5. Para todo termo M e todas variáveis x e y, $[\![\lambda x.M]\!]_\rho = [\![\lambda y.[y/x]M]\!]_\rho$, dado que $y \notin FV(M)$.
6. Para todo termo M e N, se para todo $d \in D$ tem-se que $[\![M]\!]_{[d/x]\rho} = [\![N]\!]_{[d/x]\rho}$, então $[\![\lambda x.M]\!]_\rho = [\![\lambda x.N]\!]_\rho$

$[\![M]\!]_\rho$ também pode ser escrito como $[\![M]\!]^{\mathbb{D}}_\rho$ ou simplesmente $[\![M]\!]$, quando já se sabe que a interpretação é independente de ρ.

As condições 1 - 6 imitam o comportamento que um modelo de $\lambda\beta$ precisa ter. A condição 6 fornece a interpretação no modelo da regra ξ. Porém, essas condições não são suficientes para mapear $\lambda\beta\eta$, pois elas não dizem nada sobre a η-conversão. Para isso, é necessário adicionar a seguinte definição:

Definição 1.27. Um modelo de $\lambda\beta\eta$ é um λ-modelo que satisfaz a equação $\lambda x.Mx = M$ para todo termo M e $x \notin FV(M)$.

Dada essa definição, pode-se supor que:

Teorema 1.4. Um λ-modelo $\mathbb{D}$ é extensional se, e somente se, ele é um modelo de $\lambda\beta\eta$.

1.2.3 Modelos livres de Sintaxe

O modelo definido anteriormente não define bem o que a estrutura aplicativa precisa ter como propriedades para ser um λ-modelo, já que se prende à sintaxe dos termos de $\lambda\beta$. Seria interessante definir um modelo onde não fosse necessário definir os termos antes de definir a estrutura aplicativa.

Primeiro, é necessário definir uma propriedade sobre modelos no geral:

Definição 1.28. Seja $\mathbb{D} = \langle D, \bullet, [\![]\!] \rangle$ um λ-modelo. Seja $\sim$ a *equivalência extensional* definida na definição 1.23:

$$a \sim b \iff (\forall d \in D)(a \bullet d = b \bullet d)$$

Para cada $a \in D$, a classe de equivalência extensional $\tilde{a}$ é o conjunto definido por:

$$\tilde{a} = \{b \in D : b \sim a\}$$

Para todo $a \in D$ existem M, x, ρ tais que $[\![\lambda x.M]\!]_\rho \in \tilde{a}$. Por exemplo, sejam $M \equiv ux$ e $\rho = [a/u]\sigma$ para toda valuação σ, então $\rho(u) = a$ e $[\![\lambda x.ux]\!]_\rho$ é equivalente extensionalmente a a, pois:

$$[\![\lambda x.ux]\!]_\rho \bullet d = [\![ux]\!]_{[d/x]\rho} = a \bullet d$$

Definição 1.29. (O mapeamento Λ) Seja $a \in D$ e M, x, ρ tais que $[\![\lambda x.M]\!]_\rho \in \tilde{a}$. Somente um membro de $\tilde{a}$ é igual a $[\![\lambda x.M]\!]_\rho$, esse membro será denominado de $\Lambda(a)$, onde $\Lambda : D \to D$ possui as seguintes propriedades:

1. $\Lambda(a) \sim a$
2. $\Lambda(a) \sim \Lambda(b) \iff \Lambda(a) = \Lambda(b)$
3. $a \sim b \impliedby \Lambda(a) = \Lambda(b)$
4. $\Lambda(\Lambda a) = \Lambda a$
5. Existe $e \in D$ tal que $e \bullet a = \Lambda(a)$ para todo $a \in D$.

Um desses e é o membro em D que corresponde ao numeral de Church 1, pois

$$e = [\![1]\!]_\sigma = [\![\lambda xy.xy]\!]_\sigma$$

e

$$[\![\lambda xy.xy]\!]_\sigma \bullet a = [\![\lambda y.xy]\!]_{[a/x]\sigma} = \Lambda(a)$$

Definição 1.30. (λ-modelos livres de sintaxe) Um λ-modelo livre de sintaxe é uma tripla $\langle D, \bullet, \Lambda\rangle$ onde $\langle D, \bullet\rangle$ é uma estrutura aplicativa e Λ é um mapeamento de D para D, e

1. $\langle D, \bullet\rangle$ é uma algebra combinatória (estrutura aplicativa combinatorialmente completa)
2. Para todo $a \in D$, $\Lambda(a) \sim a$
3. Para todo $a, b \in D$, se $a \sim b$, então $\Lambda(a) = \Lambda(b)$
4. Existe um elemento $e \in D$ tal que para todo $a \in D$, $\Lambda(a) = e \bullet a$

Teorema 1.5. Se $\langle D, \bullet, \Lambda\rangle$ é um λ-modelo livre de sintaxe, então é possível construir um λ-modelo $\langle D, \bullet, [\![]\!]\rangle$ definindo:

1. $[\![x]\!]_\rho = \rho(x)$, se x é uma variável
2. $[\![MN]\!]_\rho = [\![M]\!]_\rho \bullet [\![N]\!]_\rho$
3. $[\![\lambda x.N]\!]_\rho = \Lambda(a)$, onde a é qualquer elemento de D tal que $a \bullet d = [\![N]\!]_{[d/x]\rho}$ para todo $d \in D$.

De forma contrária, se $\langle D, \bullet, [\![]\!]\rangle$ é um λ-modelo então é possível construir um modelo livre de sintaxe $\langle D, \bullet, \Lambda\rangle$ definindo $\Lambda(a) = e \bullet a$, onde $e = [\![\lambda yz.yz]\!]_\rho$ para qualquer valuação ρ.

A existência de Λ pode ser caracterizada por um elemento e da seguinte forma:

Teorema 1.6. Seja $\mathbb{D} = \langle D, \bullet \rangle$ uma estrutura aplicativa tal que $\mathbb{D}$ é combinatorialmente completa e existe um elemento $e \in D$ tal que:

1. para todo $a, b \in D$, $e \bullet a \bullet b = a \bullet b$
2. para todo $a, b \in D$, se $a \sim b$, então $e \bullet a = e \bullet b$.

Então $\langle D, \bullet, \Lambda \rangle$ é um λ-modelo livre de contexto, onde $\Lambda : D \to D$ é definida por $\Lambda(a) = e \bullet a$ para todo $a \in D$.

Uma tripla $\langle D, \bullet, e \rangle$ que satisfa a hipótese do teorema anterior é chamada de λ-modelo frouxo de Scott-Meyer.

1.2.4 Ordens Parciais Completas

O modelo mais conhecido para o Cálculo λ é o Modelo de Dana Scott, o D_∞. O modelo de Dana Scott utiliza a noção de Reticulados (Lattices) Completos, mas é possível fazer uma generalização para Ordens Parciais Completas (CPOs). Alguns modelos do Cálculo λ podem ser descritos mais facilmente por CPOs do que por reticulados.

Para não precisar supor muito, é necessário voltar algumas etapas:

Definição 1.31. Seja P um conjunto. Uma *ordem*, também chamada de *ordem parcial*, em P é uma relação binária $\leq$ em P tal que, para todo $x, y, z, \in P$,

1. (Reflexividade) $x \leq x$
2. (antissimetria) Se $x \leq y$ e $y \leq x$, então $x = y$
3. (Transitividade) Se $x \leq y$ e $y \leq z$, então $x \leq z$

O par $(P, \leq)$ é chamado de *Conjunto ordenado*, ou *Poset* (Do inglês, Partially Ordered set).

Exemplos:

- O conjunto $\mathbb{N}$ dos números naturais, junto com a ordem crescente usual é um poset.
- O conjunto $\{A | A \subseteq X\}$ dos subconjuntos de um conjunto X, escrito como $\mathcal{P}(X)$ e denominado de *Conjunto Potência*, é um poset com ordem dada pela inclusão de subconjuntos $A \subseteq B$. Essa ordem é antissimetrica pois se A e A' são subconjuntos de X onde $A \subseteq A'$ e $A' \subseteq A$, então $A = A'$. Reflexividade e transitividade se seguem da mesma maneira.

Existem várias formas de mapear um conjunto ordenado em outro de forma a manter suas propriedades:

Definição 1.32. Sejam P e Q conjuntos ordenados. Um mapeamento $\phi : P \to Q$ é dito:

1. **preservante de ordem** (também chamado de **monótono**) se $x \leq y$ em P implica em $\phi(x) \leq \phi(y)$ em Q
2. **imersivo de ordem**, escrito como $\phi : P \hookrightarrow Q$, se $x \leq y$ em P se, e somente se, $\phi(x) \leq \phi(y)$ em Q

3. **isomorfismo de ordem** se é uma imersão de ordem que mapeia P em Q

Alguns conjuntos possuem um valor menor possível ou um valor maior possível, definidos da seguinte forma:

Definição 1.33. Seja P um conjunto ordenado. P possui um elemento *mínimo* se existe $\perp \in P$ tal que $\perp \leq x$ para todo $x \in P$. De forma dual, P possui um elemento *máximo* $\top \in P$ tal que $x \leq \top$ para todo $x \in P$.

Exemplos:

- O mínimo do conjunto ordenado $(\mathbb{N}, \leq)$ é o 0, mas não existe máximo.
- No conjunto ordenado $(P(X), \subseteq)$, tem-se que $\perp = \emptyset$ e $\top = X$.

Subconjuntos de conjuntos ordenados também podem possuir elementos mínimos e máximos:

Definição 1.34. Seja P um conjunto ordenado e $Q \subseteq P$. Então o elemento $u \in P$ tal que $x \leq u$ para todo $x \in Q$ é chamado de *cota superior* de Q. O elemento $l \in P$ é chamado de *menor cota superior* ou *supremo* de Q se para toda cota superior $u \in P$, $l \leq u$.
Dualmente, o elemento $u \in P$ tal que $u \leq x$ para todo $x \in Q$ é chamado de *cota inferior* de Q. O elemento $l \in P$ é chamado de *maior cota inferior* ou *infimo* de Q se para toda cota inferior $u \in P$, $u \leq l$.

Exemplo: Seja $S = \{1, 3, 5\} \subset \mathbb{N}$, então são cotas inferiores 0 e 1 e são cotas superiores todo número maior que 5.
Supremos e ínfimos podem ser tratados algebricamente da seguinte forma:

Definição 1.35.

1. O *Join* de x e y, $x \vee y$, é o supremo $sup\{x, y\}$. O supremo de um conjunto qualquer é denotado por $\bigvee S$
2. O *meet* de x e y, $x \wedge y$ é o infimo $inf\{x, y\}$. O infimo de um conjunto qualquer S é denotado por $\bigwedge S$.

Um reticulado pode ser definido por:

Definição 1.36. (Reticulado) Seja P um conjunto ordenado não vazio, então:

1. Se $x \vee y$ e $x \wedge y$ existem para todo $x, y \in P$, então P é chamado de *Reticulado*
2. Se $\bigvee S$ e $\bigwedge S$ existem para todo $S \subseteq P$, então P é chamado de *Reticulado Completo*

Definição 1.37. Um subconjunto X de P é dito *direcionado* se, e somente se, X é não vazio e para cada par de elementos $x, y \in X$, existe um elemento $z \in X$ tal que $x \leq z$ e $y \leq z$.

Agora finalmente a definição de uma ordem parcialmente completa:

Definição 1.38. Uma *Ordem Parcialmente Completa* (CPO) é um conjunto ordenado parcial $(D, \leq)$ tal que:

1. D possui um elemento mínimo
2. Todo subconjunto direcionado X de D possui um supremo. Ou seja $\bigvee X$ existe para todo $X \subseteq D$

Dessa forma, é possível ver em que medida um CPO é mais geral que um reticulado, pois ele retira a condição que o ínfimo exista para todo $X \subseteq D$.

Exemplo: Seja um objeto $\perp \notin \mathbb{N}$ e seja $\mathbb{N}^+ = \mathbb{N} \cup \{\perp\}$. Defina um ordenamento em $\mathbb{N}^+$ como:

$$a \sqsubseteq b \text{ sse } (a = \perp \text{ e } b \in \mathbb{N}) \text{ ou } a = b$$

. O par $(\mathbb{N}^+, \sqsubseteq)$ é um CPO.

É possível descrever *morfismos* entre CPOs:

Definição 1.39. Sejam D e D' cpos e $\phi : D \to D'$ uma função,

1. ϕ é chamada *monotônica* sse $a \leq b$ implica em $\phi(a) \leq' \phi(b)$
2. ϕ é chamada *contínua* sse para todo subconjunto direcionado X de D, $\phi(\bigvee X) = \bigvee \phi(X)$.

O conjunto de todas as funções contínuas entre D e D' é denotado por $[D \to D']$.

Em $[D \to D']$ é possível definir uma relação $\preceq$ tal que:

$$\phi \preceq \psi \leftrightarrow \phi(d) \leq' \psi(d) \text{ para todo } d \in D$$

Então $\preceq$ é uma ordem parcial em $[D \to D']$ e $[D \to D']$ possui um elemento final:

$$\perp(d) = \perp' \text{ para todo } d \in D$$

E, se Φ é um subconjunto direcionado de $[D \to D']$, entã opara todo $d \in D$ o conjunto $\{\phi(d) | \phi \in \Phi\}$ é um subconjunto direcionado de D'. Com isso, é possível definir uma função $\psi : D \to D'$ como:

$$\psi(d) = \bigvee \{\phi(d) | \phi \in \Phi\} \text{ para todo } d \in D$$

Então, é possível monstrar a seguinte proposição:

Proposição 1.1. Se D e D' são cpos, então $[D \to D']$ também é um cpo pelo ordenamento parcial $\preceq$ definido anteriormente. Seu último elemento é dado por $\perp'$ e para qualquer subconjunto direcionado Φ de $[D \to D']$, $\bigvee \Phi$ é uma função ψ definida como anteriormente.

Dado um cpo D_0, pode-se construir uma sequência $\{D_n\}_{n=0}^{\infty}$ de cpos definidos indutivamente como $D_{n+1} = [D_n \to D_n]$ para todo $n \geq 0$. O modelo D_∞ de Scott parte de $D_0 = \mathbb{N}^+$

Para estudar a relação das sequências $\{D_n\}_{n=0}^{\infty}$ entre si, é interessante pensar a relação de como um cpo pode estar *mergulhado* em outro.

Definição 1.40. Sejam D e D' cpos. Uma *projeção* de D em D' é um par $\langle \phi, \psi \rangle$ de funções com $\phi \in [D \to D']$ e $\psi \in [D' \to D]$ tais que:

$$\psi \circ \phi = I_D \text{ e } \phi \circ \psi \preceq I'_D$$

Onde I_D e I'_D são as funções identidade em D e D' respectivamente.

Se $\langle \phi, \psi \rangle$ é uma projeção de D' em D então ϕ mergulha D em D'.

Para entender a composição de ϕ e ψ é necessário definir o seguinte lema:

Lema 1.6. A composição de funções contínuas entre cpos é contínua. Ou seja, se D, D' e D'' são cpos e $\psi \in [D \to D']$ e $\phi \in [D' \to D'']$ e $\phi \circ \psi$ é definido por

$$\text{para todo } d \in D (\phi \circ \psi)(d) = \phi(\psi(d))$$

Então

$$\phi \circ \psi \in [D \to D'']$$

Usando a definição de $\{D_n\}_{n=0}^{\infty}$ é possível construir uma projeção $\langle \phi_n, \psi_n \rangle$ de D_{n+1} para D_n para cada n. A projeção inicial de D_1 em D_0 pode ser montada da seguinte forma: Para cada $d \in D_0$, seja κ_d uma função constante $\kappa_d(c) = d$ para $c \in D_0$. κ_d é contínua, então $\kappa_d \in [D_0 \to D_0] = D_1$. Seja $\phi_0 : D_0 \to D_1$ e $\psi_0 : D_1 \to D_0$ tais que $\phi_0(d) = \kappa_d$ para $d \in D_0$ e $\psi_0(c) = c(\perp_0)$ para $c \in D_1$ (onde $\perp_0$ é o menor elemento de D_0). ϕ_0 e ψ_0 são contínuas e é possível ver que

$$(\psi_0 \circ \phi_0)(d) = \kappa_d(\perp_0) = d = I_{D_0}$$

e

$$(\phi_0 \circ \psi_0)(f) = \kappa_d(f(\perp_0)) \preceq I_{D_1}$$

Logo $\langle \phi_0, \psi_0 \rangle$ é uma projeção de D_1 em D_0. Agora seja $\phi_n : D_n \to D_{n+1}$ e $\psi_n : D_{n+1} \to D_n$ gerados indutivamente por:

$$\phi_n(\sigma) = \phi_{n-1} \circ \sigma \circ \psi_{n-1} \text{ e } \psi_n(\tau) = \psi_{n-1} \circ \tau \circ \phi_{n-1}$$

para $\sigma \in D_n$ e $\tau \in D_{n+1}$. É possível monstrar que $\phi_n \in [D_n \to D_{n+1}]$ e $\psi_n \in [D_{n+1} \to D_n]$. Logo o par $\langle \phi_n, \psi_n \rangle$ é uma projeção de D_{n+1} em D_n.

As funções ψ_n e ϕ_n só elevam n um número por vez, então é possível definir uma função $\phi_{m,n}$ da seguinte forma:

Definição 1.41. Para qualquer $m, n \geq 0$, $\phi_{m,n} : D_m \to D_n$ é definido da seguinte forma:

$$\phi_{m,n} = \begin{cases} \phi_{n-1} \circ \phi_{n-2} \circ \cdots \circ \phi_{m+1} \circ \phi_m & \text{se } m < n \\ I_{D_n} & \text{se } m = n \\ \psi_n \circ \psi_{n+1} \circ \cdots \circ \psi_{m-2} \circ \psi_{m-1} & \text{se } m > n \end{cases}$$

Uma vez definida essa função, é possível monstrar o seguinte lema:

Lema 1.7. Sejam $m, n \geq 0$, então:

1. $\phi_{m,n} \in [D_m \to D_n]$
2. Se $m \leq n$, então $\phi_{n,m} \circ \phi_{m,n} = I_{D_m}$
3. Se $m > n$, então $\phi_{n,m} \circ \phi_{m,n} \preceq I_{D_m}$
4. Se k é um número entre m e n, então $\phi_{k,n} \circ \phi_{m,k} = \phi_{m,n}$

Prova:

1. Em $\phi_{m,n}$ existem três casos:

(a) Se $n > m$, então $\phi_{m,n} = \phi_{n-1} \circ \phi_{n-2} \circ \cdots \circ \phi_{m+1} \circ \phi_m$. É facil ver que, pelo lema da composição, sendo $\phi_m \in [D_m \to D_{m+1}]$ o inicio da cadeia de composições e $\phi_{n-1} \in [D_{n-1} \to D_n]$ o fim dessas cadeia, e sendo essa cadeia de composições contínua, então $\phi_{m,n} \in [D_m \to D_n]$

(b) Se $n = m$, $\phi_{n,n} = I_{D_n}$, mas $I_{D_n} \in [D_n \to D_n]$, logo $\phi_{n,n} \in [D_n \to D_n]$

(c) Se segue de forma análoga a (a)

2. Existem dois casos:

(a) Se $m = n$, $\phi_{n,n} \circ \phi_{n,n} = I_{D_n}$

(b) Se $m < n$, $\phi_{m,n} \in [D_m \to D_n]$ e $\phi_{n,m} \in [D_n \to D_m]$. Pelo lema da composição, $\phi_{n,m} \circ \phi_{m,n} \in [D_n \to D_n]$. O valor de $\phi_{n,m} \circ \phi_{m,n} \preceq I_{D_m}$ se segue da definição de projeção.

3. Deixado para o leitor

4. Deixado para o leitor

1.2.5 O Modelo de Scott

Uma vez feitas essas definições sobre cpos, é possível definir o modelo de Scott. Para isso, é necessário definir D_∞:

Definição 1.42.

- D_∞ é o conjunto de todas as sequências infinitas na forma
$$d = \langle d_0, d_1, d_2, \dots \rangle$$
tais que para todo $n \geq 0$ tem-se $d_n \in D_n$ e $\psi_n(d_{n+1}) = d_n$

- A relação $\sqsubseteq$ em D_∞ possui a forma:
$$d = \langle d_0, d_1, d_2, \dots \rangle \sqsubseteq \langle d'_0, d'_1, d'_2, \dots \rangle \text{ se } d_n \sqsubseteq d'_n \text{ para todo } n \geq 0$$

- Se X é um subconjunto de D_∞, então $X_n = \{a_n | a \in X\}$ é o conjunto dos n-ésimos termos de cada sequência $a \in X$.

Lema 1.8. O par $\langle D_\infty, \sqsubseteq \rangle$ definido acima é um cpo com menor elemento
$$\bot = \langle \bot_0, \bot_1, \bot_2, \dots \rangle$$
onde $\bot_n$ é o menor elemento de D_n e menor cota superior do subconjunto direcionado X de D_∞ dado por:
$$\bigvee X = \langle \bigvee X_0, \bigvee X_1, \bigvee X_2, \dots \rangle$$

Para cada $n \geq 0$ é possível definir um par de funções contínuas que formam uma projeção de D_∞ em D_n definidas como:

Definição 1.43. Para cada $n \geq 0$, seja $\phi_{n,\infty} : D_\infty \to D_n$ e $\phi_{\infty,n} : D_n \to D_\infty$ definidas por:
$$\phi_{n,\infty} = \langle \phi_{n,0}(d), \phi_{n,1}(d), \phi_{n,2}(d), \dots \rangle$$
para todo $d \in D_n$ e
$$\phi_{\infty,n}(d) = d_n$$
para todo $d \in D_\infty$

Lema 1.9. Sejam $m, n \geq 0$ com $m \leq n$ e $a, b \in D_\infty$, então

1. O par $\langle \phi_{n,\infty}, \phi_{\infty,n} \rangle$ é uma projeção de D_∞ em D_n
2. $\phi_{m,n}(a_m) \sqsubseteq a_n$
3. $\phi_{m,\infty}(a_m) \sqsubseteq \phi_{n,\infty}(a_n)$
4. $a = \bigvee_{n \geq 0} \phi_{n,\infty}(a_n)$
5. $\phi_{n,\infty}(a_{n+1}(b_n)) \sqsubseteq \phi_{n+1,\infty}(a_{n+2}(b_{n+1}))$

A parte 4 sugere que os termos a_n servem como aproximações cada vez mais certas de a em D_∞. Com isso, é possível ver a aplicação $(a_{n+1}(b_n))$ quando $n \to \infty$ como uma aproximação cada vez melhor da aplicação ab. Logo, é possível definir uma relação binária em D_∞ da seguinte forma:

Definição 1.44. Para todo $a, b \in D_\infty$,

$$a \bullet b = \bigvee \{\phi_{n,\infty}(a_{n+1}(b_n)) | n \geq 0\}$$

A autoaplicação presente no Cálculo λ pode ser implementada utilizando a aplicação $a_{n+1}(a_n)$.

Uma vez definida a relação binária, pode-se ver que o par $\langle D_\infty, \bullet \rangle$ é uma estrutura aplicativa. Pode-se definir um modelo livre de sintaxe a partir dessa estrutura. Para isso, é necessário mostrar que o par $\langle D_\infty, \bullet \rangle$ é uma algebra combinatória, ou seja mostrar que existem k e s que satisfaçam as condições da Definição 1.25.

Definição 1.45 (k_n, s_n)**.**

1. Seja $n \geq 2$. Para $a \in D_{n-1}$, $\kappa_a : D_{n-2} \to D_{n-2}$ é a função constante $\kappa_a = \psi_{n-2}(a)$ para todo $b \in D_{n-2}$. Então $k_n : D_{n-1} \to D_{n-1}$ é $k_n(a) = \kappa_a$ para todo $a \in D_{n-1}$.
2. Seja $n \geq 3$. Para $a \in D_{n-1}$ e $a \in D_{n-2}$, $\tau_{a,b} : D_{n-3} \to D_{n-3}$ é a função constante $\tau_{a,b} = a(\phi_{n-3}(c))(b(c))$ para todo $c \in D_{n-3}$ e $\sigma_a : D_{n-2} \to D_{n-2}$ tal que $\sigma_a = \tau_{a,b}$ para todo $b \in D_{n-2}$. Então $s_n : D_{n-1} \to D_{n-1}$ é $s_n(a) = \sigma_a$ para todo $a \in D_{n-1}$.

Lema 1.10.

1. Para todo $n \geq 2$, tem-se que $k_n \in D_n$ e $\psi_n(k_{n+1}) = k_n$. Logo $\psi_1(k_2) = I_{D_0} \in D_1$.
2. Para todo $n \geq 3$, tem-se que $s_n \in D_n$ e $\psi_n(s_{n+1}) = s_n$. Logo $\psi_1(\psi_2(s_3)) = I_{D_0} \in D_1$.

Agora finalmente pode-se definir k e s:

Definição 1.46. Sejam k e s as seguintes sequências:

$$k = \langle \bot_0, I_{D_0}, k_2, k_3, \dots \rangle \text{ e } s = \langle \bot_0, I_{D_0}, \psi_2(s_3), k_3, k_4, \dots \rangle$$

Lema 1.11. As sequências k e s são elementos de D_∞

Lema 1.12. Para todo $a, b, c \in D_\infty$, $k \bullet a \bullet b = a$ e $s \bullet a \bullet b \bullet c = a \bullet c \bullet (b \bullet c)$

Logo, pelo lema anterior, é possível ver que o par $\langle D_\infty, \bullet \rangle$ é uma álgebra combinatória. O que falta para provar que esse par é um modelo é monstrar que ele é extensional.

Lema 1.13. $\langle D_\infty, \bullet \rangle$ é uma álgebra combinatória extensional

Prova: Sejam a e b elementos de D_∞ tais que $a \sim b$, ou seja, $a \bullet c = b \bullet c$ para todo $c \in D_\infty$. Seja $m \geq 0$ e d um elemento arbitrário de D_m. Seja $c = \phi_{m,\infty}(d)$. Pode ser provado que $(a \bullet c)_m = a_{m+1}(d)$ e $(b \bullet c)_m = b_{m+1}(d)$. Logo $a_{m+1} = (a \bullet c)_m = (b \bullet c)_m(d) = b_{m+1}$ e $a_{m+1} = b_{m+1}$. Ou seja $a_n = b_n$ para $n > 0$ e $a_0 = \psi_0(a_1) = \psi_0(b_1) = b_0$ (Pois ψ é contínua), logo $a = b$. Logo, $\langle D_\infty, \bullet \rangle$ é extensional.

Com isso, fica provado que $\langle D_\infty, \bullet \rangle$ é um λ-modelo livre de sintaxe