Teoria da Computação
Definições Indutivas, Conjunto Estruturado de Termos, Demonstração por Indução Estrutural, Programação por Recursão Estruturada

Simão Melo de Sousa

1. Aviso Prévio

- A redacção deste documento baseou-se fortemente na bibliografia indicada. Parece-nos então óbvio que a leitura e a aprendizagem directa pelas obras originais é recomendada, e mesmo essencial à compreensão profunda das noções aqui apresentadas;
- O português não é a língua materna do autor e o presente documento encontra-se em fase de elaboração pelo que se agradece e até se incentiva qualquer sugestão ou correcção.

2. Bibliografia

Consultar [1, 2, 3, 4, 6, 5]

2.0.0.1. Contexto O conjunto das noções que introduzimos neste texto baseam-se essencialmente na modelização "formal" e finita dos objectos que se pretende estudar, quer esses sejam matemáticos, informáticos ou mesmo reais. Um contributo importante, explorado inicialmente pelo lógico E. Post nos anos 40, para essa modelização são as definições indutivas a partir das quais podemos definir uma classe importante dos objectos que pretendemos aqui estudar. Este estado de facto justifica uma descrição dos mecanismos envolvidos nas definições indutivas assim como das propriedades essenciais que se extraem de tais definições.

3. Indução Estrutural

A informática, assim como parte da matemática, é baseada na representação do mundo real por entidades formais nas quais se exerce o raciocínio ou a programação. Este processo de modelização formal de entidades reais, intuitivas, ou não completamente conhecidas revelou-se, ao longo da história da matemática fonte de verdadeiras revoluções na matemática. O processo de capturar o mundo à nossa volta pela formalização, levou à construção de sistemas axiomáticos. Um deles, o famoso "sistema axiomático da aritmética de Peano", propõe uma construção dos naturais, ou seja, um método construtivo para a obtenção dos inteiros. Os inteiros, embora estejam em número infinito, são vistos como construções finitas, baseadas em regras bem estabelecidas, aos quais se pode submeter, por isso mesmo, certos tipos de raciocínio.

De forma mais abrangente consegue-se definir objectos, conjuntos, funções a partir de outros estabelecendo regras de construção que se baseiam na estrutura que queremos dar aos objectos que pretendemos construir. Assim a noção fundamental neste tipo de definição é a estrutura interna que se salientou no objecto construído. Classicamente, essas definições de conjuntos são chamadas **definições indutivas**, o tipo de raciocínio que elas permitem é referido por **indução estrutural** e as funções cuja definição tira proveito da estrutura de tais conjuntos são referidas de **funções estruturalmente recursivas**.

Assim o primeiro contributo fundamental dessa abordagem é a possibilidade de modelar objectos finitos. Outro contributo, já referido, é a possibilidade de prova por métodos simples, elegantes e eficazes.

Uma justificação do estudo dessas definições é que os objectos que modelam as definições indutivas são exactamente os que o computador é capaz de modelar. Um computador só trabalha com objectos finitos, e se tem de manipular objectos infinitos, só o faz à custa de representações/aproximações finitas. Assim a modelação indutiva é constantemente usada em estruturas de dados assim como na elaboração de programas em linguagens funcionais ou lógicas. A prova de tais programas, actividade altamente importante na informática de hoje, contenta-se em seguir os esquemas recursivos herdados pelas boas propriedades deste tipo de definições.

Veremos também que, por querer representar o mundo à nossa volta, precisamos de uma linguagem. Raciocinar consiste num discurso nesta linguagem, mas permanece no

Página Pessoal
Página de Rosto
◀◀ ▶▶
◀ ▶

Retroceder
Ecrã Todo
Fechar
Sair

entanto o problema da capacidade da linguagem para representar os objectos do discurso. Tanto a lógica como as suas representações computacionais, como o cálculo λ (λ-calculus na sua terminologia original), têm de disponibilizar meios de definição de objectos, além dos tipos de objectos propostos "por defeito". Em termos de programação isso corresponderia à disponibilização de ferramentas de definição de registos, vectores, (etc, ...), tão úteis a essa actividade. Programar só com inteiros ou caracteres limita muito esta actividade. Em matemática dispor de meios *construtivos* para representar um conjunto particular simplifica grandemente à sua utilização. Em lógica essas ferramentas são classicamente as definições indutivas. Veremos mais tarde que os objectos definidos por indução tem a propriedade de ser tratáveis computacionalmente, ou seja, pertencem ao tipo dos objectos **computáveis**.

Vamos nas secções seguintes esclarecer formalmente a noção de definição indutiva, estabelecer as propriedades dos objectos definidos desta forma, assim como mostrar quais são os princípios de indução que emergem de tais definições. Para ilustrar essas noções tomaremos como exemplo dois casos particulares, o conjunto dos naturais e os conjuntos de termo.

3.1. Conjuntos definidos indutivamente

Definição 3.1 (Conjuntos definidos indutivamente) *Seja E um conjunto. Uma definição indutiva de uma parte X de E consiste nos dois dados seguintes:*

- *num subconjunto não vazio B de E, chamado* **base**,
- *num conjunto K de operações $\phi : E^{\sharp(\phi)} \longrightarrow E$, onde $\sharp(\phi)$ representa a cardinalidade de ϕ* [1].

X é então definido como o menor conjunto que verifica as asserções seguintes:

(B): $B \subseteq X$,

(I): Para todo o $\phi \in K$, para todos os $x_1, \ldots, x_{\sharp(\phi)} \in X, \phi(x_1, \ldots, x_{\sharp(\phi)}) \in X$.

[1] Chamaremos, por vezes neste texto, aridade à cardinalidade $\sharp(\phi)$.

O conjunto definido é então

$$X = \bigcap_{Y \in \mathcal{F}} Y$$

onde

$$\mathcal{F} = \{Y \subseteq E / B \subseteq Y \text{ e } Y \text{ verifica } (I)\}$$

Por razões de conveniência, poderemos usar a notação alternativa seguinte para as definições indutivas:

(B): $x \in X \quad$ *(para todo o* $x \in B$*),*

(I): $x_1, \ldots, x_{\sharp(\phi)} \in X \Longrightarrow \phi(x_1, \ldots, x_{\sharp(\phi)}) \in X \quad$ *(para todo o* $\phi \in K$*).*

3.1.0.2. Considerações técnicas. Vamos, numa primeira fase e para simplificar, considerar que o conjunto K é finito e que $\sharp(\phi) \in \mathbb{N}$. As definições indutivas obtidas sem essas hipóteses iniciais serão estudadas na secção 3.4

Finalmente, para evitar algumas definições triviais sem interesso, vamos impor que o conjunto K nunca seja vazio.

3.1.0.3. Conjuntos verificando (B) e (I): É de salientar que, em geral, vários conjuntos verificam (B) e (I), e que esse facto nos obriga a ter um certo cuidado com os elementos do conjunto desejado, realçando assim a importância dada a propriedade de menor conjunto: Consideramos por exemplo a definição indutiva de uma parte P de $\mathbb{N}$:

(B): $0 \in P$,

(I): $n \in P \Longrightarrow n + 2 \in P$.

Existe uma infinidade de partes de $\mathbb{N}$ que verificam (B) e (I); $\mathbb{N}$, $\mathbb{N} \setminus \{1\}$, $\mathbb{N} \setminus \{1, 3, 5, \ldots\}$ são exemplos dessas partes. Mas a parte P definida indutivamente (isto é a menor parte de $\mathbb{N}$ verificando as duas asserções) não é nem mais nem menos do que o conjunto dos naturais pares.

3.1.0.4. O menor conjunto verificando (B) e (I): O conjunto $\mathcal{F}$ não é vazio porque E pertence a $\mathcal{F}$. Isto porque E verifica (B) ($B \subseteq E$) e (I) trivialmente.

Além disso, se partes de um conjunto verificam uma condição então a intersecção dessas partes também a verifica. Seja $\mathcal{Y}$ um conjunto de partes Y de E verificando (B) e (I), e seja $Z = \bigcap_{Y \in \mathcal{Y}} Y$; por B estar incluído em todo o conjunto Y pertencendo a $\mathcal{Y}$, B está incluído em $Z = \bigcap_{Y \in \mathcal{Y}} Y$, logo Z verifica (B); se $x_1, \ldots, x_{\sharp(\phi)} \in Z$, então para todo o $Y \in \mathcal{Y}$, $x_1, \ldots, x_{\sharp(\phi)} \in Y$, donde $\phi(x_1, \ldots, x_{\sharp(\phi)}) \in Y$, e então $\phi(x_1, \ldots, x_{\sharp(\phi)}) \in Z$ e Z verifica (I). Resumindo, $\underline{\bigcap_{Y \in \mathcal{F}} Y}$, onde $\mathcal{F}$ é o conjunto das partes de E definido como referido previamente, <u>é, então, a mais pequena parte de E verificando (B) e (I),</u> isto é, <u>o conjunto indutivo definido por (B) e (I)</u>.

Exemplo 3.1 *1. A parte X de $\mathbb{N}$ definida indutivamente por*

(B): $0 \in X$,
(I): $n \in X \Longrightarrow \mathit{successor}(n) \in X$.

não é nem mais nem menos do que $\mathbb{N}$.

2. Seja $A = \{(,)\}$ o alfabeto constituído por dois parêntesis, o conjunto $D \subseteq A^*$ das palavras bem formadas a partir do alfabeto A, chamado a linguagem de Dyck é definida por:

(B): $\epsilon \in D$,
(I): (a) $x \in D \Longrightarrow (x) \in D$.
(b) $x, y \in D \Longrightarrow xy \in D$.

3. O conjunto AB das árvores binárias etiquetados sobre o alfabeto A é a parte de $(A \cup \{\emptyset, "(", ")", ","\})^*$ definida indutivamente por [2]

(B) $\emptyset \in AB$ *(é a árvore vazia)*
(I) $g, d \in AB, \forall a \in A \Longrightarrow (a, g, d) \in AB$ *(a árvore de raíz a e de filho esquerdo g e de filho direito d).*

[2] Ver secção 4 para uma introdução aos conceitos de alfabeto, palavra, sequência de caracteres, ∗...

4. *O conjunto E das expressões da aritmética elementar: Seja $\mathcal{V}$ um conjunto numerável de variáveis, o conjunto das expressões aritméticas com parêntesis formada a partir dos inteiros e das variáveis de $\mathcal{V}$ é a parte de $(\mathcal{V} \cup \mathbb{N} \cup \{"(","\,)","+","-","*","\,/"\})^*$ definida indutivamente por*

$$
\begin{array}{lll}
(B): & (a) & \forall n \in \mathbb{N}, n \in E \\
 & (b) & \forall v \in \mathcal{V}, v \in E \\
(I): & (a) & e \in E \Longrightarrow +e \in E \\
 & (b) & e \in E \Longrightarrow -e \in E \\
 & (c) & e_1, e_2 \in E \Longrightarrow (e_1 + e_2) \in E \\
 & (d) & e_1, e_2 \in E \Longrightarrow (e_1 - e_2) \in E \\
 & (e) & e_1, e_2 \in E \Longrightarrow (e_1 * e_2) \in E \\
 & (f) & e_1, e_2 \in E \Longrightarrow (e_1 / e_2) \in E
\end{array}
$$

Este último exemplo ilustra o aspecto contractivo das definições de gramáticas e sintaxes BNF. De facto a quase totalidade das definições de gramáticas podem ser vistas como definições indutivas. Por exemplo podemos definir indutivamente o conjunto dos programas que contem só atribuições e ifs:

5. *Sejam E o conjunto das expressões válidas (que pode ser definido de forma análoga ao E do ponto anterior), C o conjunto das expressões condicionais válidas ($C \subseteq E$) e $\mathcal{V}$ um conjunto de variáveis numeráveis. A parte P de $(E \cup C \cup \{ :=, if, then, else, begin, end, ; \})^*$ dos programas com alternativa é definido indutivamente por*

$$
\begin{array}{lll}
(B): & \multicolumn{2}{l}{\forall v \in \mathcal{V} \text{ e } \forall e \in E, v := e; \in P,} \\
(I): & (a) & c \in C \text{ e } p_1, p_2 \in P \Longrightarrow if\ c\ then\ begin\ p_1\ end;\ else\ begin\ p_2\ end;\ \in P \\
 & (b) & c \in C \text{ e } p \in P \Longrightarrow if\ c\ then\ begin\ p\ end;\ \in P \\
 & (c) & p_1, p_2 \in P \Rightarrow p_1 p_2 \in P
\end{array}
$$

O resultado, que vamos a seguir enunciar, ilustra a qualidade construtiva de uma definição indutiva: um elemento de uma parte X de E definida indutivamente é obtida por um processo finito que usa as propriedades da definição de X.

Teorema 3.1 (Construção de elementos de um conjunto indutivo) *Se X é definido pelas condições (B) e (I), então qualquer elemento de X pode ser obtido a partir da base B aplicando um número finito de vezes o passo (I).*

Demonstração:
Seja

$$\begin{array}{rcl} X_0 & = & B \\ X_{n+1} & = & X_n \cup \{\phi(x_1, \ldots, x_{\sharp(\phi)})/ \ x_1 \cdots x_{\sharp(\phi)} \in X_n \ e \ \phi \in K\} \end{array}$$

Mostra-se numa primeira parte que $\forall n \in \mathbb{N}, X_n \subseteq X$ e deduz-se que $X_\omega = \bigcup_{n\in\mathbb{N}} X_n \subseteq X$.
$X_0 = B$ por isso $X_0 \subseteq X$. Formulamos agora a hipótese que $X_n \subseteq X$ e verificamos que temos então $X_{n+1} \subseteq X$.
Seja ϕ um dos operadores de K ($\phi \in K$). Sejam $x_1, \ldots, x_{\sharp(\phi)} \in X_n$, $x_1 \ldots, x_{\sharp(\phi)} \in X$ por hipótese. Aplicando o passo indutivo (I) de X obtemos que $\phi(x_1, \ldots, x_{\sharp(\phi)}) \in X$.
Consequentemente $\{\phi(x_1, \ldots, x_{\sharp(\phi)})/ \ x_1 \cdots x_{\sharp(\phi)} \in X_n \ e \ \phi \in K\} \subseteq X$, logo $X_{n+1} \subseteq X$.
Concluí-se nesta primeira parte da prova que $X_\omega \subseteq X$.
X_ω é exactamente o conjunto dos elementos que se obtém a partir de elementos da base e aos quais se aplica um número finito (≥ 0) de passos de (I).
Mostramos numa segunda parte que $X \subseteq X_\omega$, ou seja que X_ω verifica (B) e (I).
$B = X_0 \subseteq X_\omega$, logo (B) é verificado por X_ω.
Seja $\phi \in K$ e $x_1, \ldots, x_{\sharp(\phi)} \in X_\omega$. cada x_i (tal que $1 \leq i \leq \sharp(\phi)$) pertence a um $X_{n_i} \subseteq X_\omega$.
Seja $n = max(n_1, \ldots, n_{\sharp(\phi)})$ então $x_i \in X_n$ então $\phi(x_1, \ldots, x_{\sharp(\phi)}) \in X_{n+1} \subseteq X_\omega$. Por isso X_ω verifica (I). Logo $X \subseteq X_\omega$.
<u>Concluímos então que $X = X_\omega$.</u>
$\diamond$

Usando os conjuntos definidos na demonstração deste teorema, define-se a noção muito útil de **altura** de um elemento x de X como sendo o menor n tal que $x \in X_n$.

3.2. Provas por indução estrutural

O resultado 3.1 que acabamos mostrar permite descriminar os elementos de X dos outros elementos de E (ou seja $E \setminus X$). Eles são os que tem a estrutura indicada pela definição.

Uma propriedade estabelecida sobre esse conjunto caracterizado de tal forma, terá necessariamente uma validação com base nessa mesma estrutura.

Assim a proposição seguinte mostra que uma propriedade que conserva a sua validade após os passos da indução, ou seja que é validada pela estrutura de X, é então válida para todos os elementos x de X.

Teorema 3.2 (Indução Estrutural) *Seja X um conjunto indutivo (ver definição 3.1) e seja $P(x)$ uma asserção exprimindo uma propriedade de um elemento x de X. se se verificar:*

(B') Para cada $x \in B$, $P(x)$ é válida.

(I') Para cada $\phi \in K$ $(P(x_1), \ldots, P(x_{\sharp(\phi)})) \Longrightarrow P(\phi(x_1, \ldots, x_{\sharp(\phi)}))$

então $P(x)$ é válida para todo o $x \in X$.

Assim, verificar (B') e (I') para uma propriedade P em X representa uma prova por indução estrutural de P no conjunto X.

Demonstração:
Seja Y o conjunto dos x tais que $P(x)$ seja válida. Basta-nos provar que $Y \supseteq X$. Temos $B \subseteq Y$ (por (B')) e Y verifica (I) a custa de (I'). Por isso $Y \supseteq X$.
$\diamond$

3.3. Raciocínio por indução sobre $\mathbb{N}$

O princípio de indução enunciado é uma generalização do princípio de indução sobre $\mathbb{N}$, definido por exemplo na aritmética de Peano. De facto, Peano define os inteiros naturais como sendo o conjunto indutivamente gerado a partir de *zero* e da função *sucessor*. Assim como consequência do teorema 3.2 temos o princípio de indução, sobre os naturais, seguinte:

Teorema 3.3 (Primeiro princípio de indução sobre os naturais) *Seja* $P(x)$ *uma propriedade dependendo de um* $x \in \mathbb{N}$*. se* P *verificar:*

($B_{\mathbb{N}}$*)* $P(0)$ *é válida.*

($I_{\mathbb{N}}$*) Para todo o* $n \in \mathbb{N}$ $(P(n) \Longrightarrow P(n+1))$

então $P(x)$ *é válida para todo o* $x \in \mathbb{N}$*.*

3.4. Considerações técnicas

Vamos nesta secção revisitar a noção de conjunto indutivo no seu contexto geral (i.e. sem as hipóteses iniciais simplificadoras) realçando as ligações entre conjuntos indutivos, conjuntos fechados e pontos fixos de operadores. Consideraremos também aspectos notacionais.

Definição 3.2 (Regra)
Uma regra é um par (X, x) *onde* X *é um conjunto, chamado conjunto de premisses, e* x *é a conclusão.*

Existem notações alternativas para a definição de conjuntos indutivos: a notação por conjunto de regras e a notação por regras de inferências. A primeira alternativa é de facto a notação original, como a encontramos em [1], e realça a natureza formal dos conjuntos indutivos definidos. Assim a regra (X, x) é representada por $X \longrightarrow x$. Pelo símbolo $\longrightarrow$ é realçada a noção de produção: dos elementos de X produz-se o elemento x. A segunda notação é mais adequada para conjuntos utilizados em contextos lógicos, como acontece na definição de sistemas dedutivos em que a noção de inferência é central. Neste enquadramento é utilizada a representação $\dfrac{X}{x}$ para a regra (X, x): das premissas $\phi \in X$ deduz-se/infere-se a conclusão x. Por serem ambas equivalentes, utilizaremos sem discriminação estas duas notações.

Uma definição indutiva é então dada por um conjunto de regras. Os conjuntos B e I são, de facto, contemplados nesta reformulação das definições indutivas. Seja Φ um conjunto de regras, B é o subconjunto de Φ das regras (X, x) tais que $X = \emptyset$. O conjunto I é o conjunto das regras restantes, ou seja $I = \Phi - B$.

Seja Φ um conjunto de regras, utilizaremos a notação $\Phi : X \longrightarrow x$ para denotar que $X \longrightarrow x$ pertence a Φ.

Nenhuma condição é imposta aos conjuntos Φ e X. Estes podem ser vazio ou até mesmo infinito. Notemos simplesmente que uma definição indutiva em que Φ é vazio é pouco interessante em si. Designaremos por regras finitas as regras (X, x) em que X é finito.

Como caracterizar nesta formulação as noções de conjunto indutivo gerado por um conjunto de regras, princípio de indução ou ainda de construção de elementos indutivos?

A resposta, de facto já referida, passa pelas noções de ponto fixo ou de conjunto fechado.

3.4.0.5. Caracterização em termos de conjunto fechado

Definição 3.3 (Conjuntos indutivos e conjuntos fechados)

- *Se Φ é um conjunto de regras, então um conjunto A é fechado por Φ se cada regra de Φ que tenha as suas premisses em A também tenha a conclusão em A. Formalmente temos: A é fechado por Φ se $\forall(\Phi : X \longrightarrow x).\ X \subseteq A \rightarrow x \in A$*

- *Se Φ é um conjunto de regras então o conjunto indutivamente definido por Φ, notado $I(\Phi)$, é definido por $I(\Phi) \triangleq \bigcap\{A \mid A \text{ é fechado por } \Phi\}$.*

O princípio de indução associado a um conjunto $I(\Phi)$ define-se então da seguinte forma: Se P é uma propriedade tal que para toda regra $\Phi : X \longrightarrow x\ \forall y \in X.P(y)$ implica que $P(x)$, então P é verificado por todos os elementos de $I(\Phi)$. Dito por outras palavras, para verificar que uma determinada propriedade P é verificado por um conjunto indutivo $I(\Phi)$ basta verificar que o conjunto $\{x \mid x \in I(\Phi) \wedge P(x)\}$ é Φ-fechado.

Para o caso em que Φ só conter regras finitas, podemos reformular a noção de construção de elementos de um conjunto indutivo em termos de Φ-prova.

Definição 3.4 (Φ-prova)
A sequência $a_0, \ldots, a_n$ é uma Φ-prova (finita) de b se

- $a_n = b$;

- *Para todo o* $m \leq n$ *existe um* $X \subseteq \{a_i \mid i < m\}$ *tal que* $\Phi : X \longrightarrow a_m$.

O conceito de Φ-prova fica ainda mais explícito se utilizarmos a notação por regras de inferências. Neste caso uma Φ-prova é uma árvore de derivação em que as folhas são $\emptyset$ (as regras aí utilizadas são as regras de I) e a conclusão b é a raíz.

Por exemplo, se consiredarmos o conjunto E das expressões aritméticas elementares, Φ é o conjunto infinito de regras finitas seguintes:

$$
\begin{array}{l}
(\emptyset, n),\ \textit{para todo o } n \in \mathbb{N} \\
(\emptyset, x),\ \textit{para todo o } x \in \mathcal{V} \\
(\{e\}, +e) \\
(\{e\}, -e) \\
(\{e_1, e_2\}, (e_1 + e_2)) \\
(\{e_1, e_2\}, (e_1 - e_2)) \\
(\{e_1, e_2\}, (e_1 * e_2)) \\
(\{e_1, e_2\}, (e_1 / e_2))
\end{array}
$$

é alternativamente notado por

$$
\begin{array}{l}
\emptyset \longrightarrow n, \textit{para todo o } n \in \mathbb{N} \\
\emptyset \longrightarrow x,\ \textit{para todo o } x \in \mathcal{V} \\
\{e\} \longrightarrow +e \\
\{e\} \longrightarrow -e \\
\{e_1, e_2\} \longrightarrow (e_1 + e_2) \\
\{e_1, e_2\} \longrightarrow (e_1 - e_2) \\
\{e_1, e_2\} \longrightarrow (e_1 * e_2) \\
\{e_1, e_2\} \longrightarrow (e_1 / e_2)
\end{array}
$$

ou por

$$\frac{}{n},\ para\ todo\ o\ n \in \mathbb{N}$$

$$\frac{}{x},\ para\ todo\ o\ x \in \mathcal{V}$$

$$\frac{e}{+e}$$

$$\frac{e}{-e}$$

$$\frac{e_1 \quad e_2}{(e_1 + e_2)}$$

$$\frac{e_1 \quad e_2}{(e_1 - e_2)}$$

$$\frac{e_1 \quad e_2}{(e_1 * e_2)}$$

$$\frac{e_1 \quad e_2}{(e_1 / e_2)}$$

e os seguintes exemplos são Φ-provas de $((5 + x)/n)$

$$\frac{\dfrac{\dfrac{}{5} \qquad \dfrac{}{x}}{(5+x)} \qquad \dfrac{}{n}}{((5+x)/n)}$$

e

$$5,\ x,\ n,\ (5 + x),\ ((5 + x)/n)$$

Proposição 3.1 (Φ-provas e conjunto indutivo)
No caso de Φ ser um conjunto de regras finitas, temos $I(\Phi) = \{b \mid b\ tem\ uma\ \Phi - prova\}$

3.4.0.6. Teorias, Provas e Teoremas Uma pergunta que nos podemos colocar é a da relevância dessas definições alternativas às noções de conjunto indutivo, de elemento dum conjunto indutivo e das suas derivações.

Tais noções são, de facto, centrais na matemática. As teorias que a fundamentam são definidas como conjuntos de regras. Identificamos assim uma teoria T com a sua formulação em termos de conjunto de regras. Por exemplo regras da seguinte forma $(\emptyset, x)$ (ou $\frac{}{x}$) são os axiomas da teoria considerada. Regras como (X, x) (ou $\frac{X}{x}$) representam, neste contexto, regras de dedução da teoria: se as condições X são reunidas (ou, dito de outra forma, se são verificadas as propriedades X) então x é verificado. Digamos neste caso que x é um teorema. A demonstração do teorema x na teoria ϕ é então *exactamente* o que definimos como ϕ-prova. Utilizamos então a notação $\phi \vdash x$ para designar que x é um teorema da teoria ϕ, ou dito de formas alternativas, que existe uma ϕ-prova de x, que x é elemento do conjunto indutivo gerado por ϕ.

Esta proximidade de conceitos é de tal forma estabelecida que se utiliza sem distinção este vocabulário e notações. Assim quando falamos de teorias falamos de conjuntos indutivos, de demonstrações falamos de ϕ-provas, de teoremas falamos de elementos de conjuntos indutivos.

3.4.0.7. Caracterização em termos de ponto fixo A um conjunto de regras Φ podemos associar o operador monótono (de facto contínuo) F sobre as partes dum universo $\mathcal{U}$ servindo de base a definição de conjuntos indutivos.

$$F(X) \triangleq \{x \in \mathcal{U} \mid \exists Y \subseteq X.(Y, x) \in \Phi\}$$

De forma semelhante podemos associar a qualquer operador monótono F um conjunto de regras Φ

$$\Phi \triangleq \{(X, x) \mid X \subseteq \mathcal{U} \wedge x \in F(X)\}$$

Estas duas operações estão em bijeção e reparamos que se Φ e F se assemelham então temos ao facto seguinte: Qualquer conjunto A é fechado por Φ se e só se $F(A) \subseteq A$.

Dito por outras palavras, o conjunto indutivo definido pelo conjunto de regras Φ é o menor ponto fixo do operador F associado.

3.4.0.8. indução estrutural e indução bem fundada Uma consequência interessante das regras infinitas é a possibilidade de poder unificar indução bem fundade e indução estrutural. Podemos expressar a indução bem fundada em termos de indução estruturada e *vice-versa*.

De facto se consideramos uma relação bem fundada $>$ sobre um conunto A podemos então construir o conjunto de regras $\Phi_<$ seguinte:

$$\Phi_< \triangleq \{(\{x \in A \mid x < a\}, a)\}$$

Nestas condições, o princípio de indução bem fundada induzida por $<$ confunde-se com o princípio de indução estrutural de $\Phi_<$, como o estabelece a seguinte proposição.

Proposição 3.2

$$> \;=\; I(\Phi_<)$$

Por outro lado, dado um conjunto de regras Φ, podemos definir uma relação $<$ seguinte:

$$\text{para toda a regra } \Phi : (X, a) \text{ e para todo o element x} \in X, x < a$$

Temos então o resultado seguinte:

Proposição 3.3 *A relação $<$ é bem fundada.*

Nessas condições a indução estruturada assemelha-se como um caso particular de indução bem fundada. Veremos em particular, na secção seguinte, que um bom candidato para a relação $<$ no contexto dos termos é a relação de subtermo.

4. Termos

Até agora construímos conjuntos a partir de conjuntos. Definir o conjunto dos naturais indutivamente a partir dos naturais pode parecer artificial [3]. Mas ganhamos as boas propriedades que acabamos de enunciar nas secções anteriores.

Existe no entanto uma classe de definições indutivas que nos permite em parte resolver esta problemática. Esta classe é baseada sobre um conjunto "abstracto": o conjunto dos termos [4].

Dado um alfabeto A, um conjunto de termos é um subconjunto do monóide livre A^*. Desta forma, um termo é uma palavra construída a partir de um alfabeto A. O conjunto A^* das palavras é, relembremos, o conjunto de todas sequência de letras do alfabeto, a palavra "vazia" incluída. Assim, para todo o termo t, $t \in A^*$. Mas cuidado, a reciproque não é verdade, uma palavra t pertencendo a A^* não é necessariamente um termo [5].

Definir uma parte indutiva de A^* é definir um conjunto de termos que possuem uma estrutura comum, que se submetem a uma dada *gramática*: isto é, que pertencem a uma dada *linguagem*.

Ora, bem sabemos que uma linguagem não passa de algo abstracto, de um conjunto de sons ou de letras, que permite representar entidade reais. Também sabemos que a linguagem funciona porque a cada palavra se associou um ou vários significados, se associou uma ou várias entidades reais. Assim a palavra "cão" está associada ao animal peludo de quatro patas que ladra e que gosta de roer ossos [6] [7]. Assim procederemos com o conjunto dos

[3] Por exemplo na tentativa de recriar a noção de conjunto...

[4] Ver discussão em [1] sobre as definições indutivas e os fundamentos da matemática.

[5] No alfabeto usual, a sequência "aaaaa" é palavra mas não tem significado em português (não é um termo da língua portuguesa, não pertence ao conjunto dos termos portugueses).

[6] Uma nota exótica deste estado de facto pode ser uma das obras do pintor e artista Magritte que *representou* num dos seus quadros um cachimbo. Lê-se neste mesmo quadro em legenda *Ceci n'est pas une pipe* - isto não é um cachimbo. Efectivamente, trata-se de uma representação (desenho) de um cachimbo que associamos imediatamente ao objecto cachimbo. De forma geral, esta identificação abusiva não tem importância, mas há de perceber que existe uma distinção, esta sim, importante em lógica.

[7] Outra nota deriva do facto de uma linguagem poder ser vista como um conjunto de associação palavras-significados. Duas linguagens podem ter várias palavras em comum mas os significados dessas mesmas palavras não são necessariamente os mesmos. Experimente por exemplo dizer a um espanhol que gosta muito de Braga. Há de, então, ter um cuidado por não cair nestes erros no estudo das definições indutivas.

Aviso Prévio
Bibliografia
Contexto
Indução Estrutural
Termos
Funções definidas por . . .
Aplicações

Página Pessoal
Página de Rosto
Página 18 de 32
Retroceder
Ecrã Todo
Fechar
Sair

termos.

Um elemento tipo deste conjunto pode ser **interpretado** como o representante de uma entidade "concreta", da mesma forma que a palavra "zero" representa (é interpretado como) o inteiro 0.

A actividade de modelação informática tira proveito desta abordagem. De facto a actividade da programação consiste na elaboração de um discurso numa determinada linguagem. Assim representa-se a custa dos termos muitas estruturas informáticas com as quais o programador trabalha.

4.1. Definições

Seja $F = \{f_0, f_1, \ldots, f_n, \ldots\}$ um conjunto de símbolos de operações. A cada símbolo f está associado uma cardinalidade $\sharp(f) \in \mathbb{N}$ que representa o número de argumentos de f.

Seja U o conjunto de todas as sequências de símbolos pertencendo a $F \cup \{"(", ")", ","\}$.

$$U = (F \cup \{"(", ")", ","\})^*$$

Seja F_i o conjunto dos símbolos de cardinalidade i.

Definição 4.1 (Conjunto dos termos de F) *O conjunto T dos termos construidos sobre F é definido indutivamente por*

(B) $B = F_0$;

(I) Para todo o $f \in F_n$ *de cardinalidade* n, *para* $t_1 \cdots t_n \in T \Longrightarrow \phi_f(t_1, \ldots, t_n) = f(t_1, \ldots, t_n) \in T$ [8]

Convém, no seguimento desta definição, estabelecer ou relembrar algum vocabulário útil a compreensão dos resultados que seguirão. Designa-se por **comprimento** de um termo

[8] A notação utilizada para a representação dos termos, a representação prefixa *f(x,y)*, não é única mas é a mais geral. No entanto, em alguns casos, outros tipos de notações poderão ser utilizadas, como por exemplo a notação infixa *(x f y)* nos formalismos lógicos ou aritméticos em que f é uma operação binária.

Aviso Prévio
Bibliografia
Contexto
Indução Estrutural
Termos
Funções definidas por . . .
Aplicações

Página Pessoal
Página de Rosto
Página 19 de 32
Retroceder
Ecrã Todo
Fechar
Sair

t o número de caracteres da palavra, assim como se designa por **concatenação** de duas palavras de U, u e v, a operação que consiste em obter uma nova palavra w pela junção de v a u. A nova palavra w é usualmente notada uv. Uma palavra u é designada **segmento inicial** de w se existir uma palavra v tal que $w = uv$. u é designada de **segmento inicial próprio** se u não for nem a palavra vazia nem w.

Exemplo 4.1 *Seja* $F_0 = \{a\}$ *e* $F_1 = \{s\}$, $F = F_0 \cup F_1$.

- $a,\ s(a),\ s(s(a)),\ s(a(s)),\ s(as),\ s(s$ *são palavras de* U^*
- $a,\ s(a),\ s(s(a))$ *são palavras de* U^* *que são termos, isto é, que pertencem a* T.
- $s(s$ *é um segmento inicial (próprio) de* $s(s(a))$.

4.1.0.9. Representação em árvore Um termo $f(t_1, \ldots, t_n)$ pode ser representado de forma cómoda por uma árvore em que a raíz é f e em que os n filhos, da raíz, são as árvores representando os termos $t_1, \cdots, t_n$.

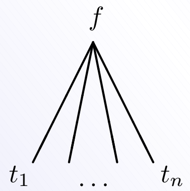

Figure 1: Árvore representando o termo $f(t_1, \ldots, t_n)$

É de realçar aqui que a noção de altura de um termo t coincide aqui com a noção da altura da árvore representando t.

4.2. Interpretação dos termos

Passamos agora a descrever os mecanismos de associação de um termo a um objecto "concreto" que o termo passará a representar.

Seja V um conjunto. A cada elemento f de F_0 associa-se um elemento $h(f)$ de V. A cada elemento f de F_i, com $i > 0$, associa-se uma aplicação $h_f : V^i \longrightarrow V$.

Proposição 4.1 (Interpretação de termos) *Existe uma e uma só função h^* de T para V tal que:*

(B_h) Se $t \in F_0$ então $h^*(t) = h(t)$;

(I_h) Se $t = f(t_1, \ldots, t_n)$ então $h^*(t) = h_f(h^*(t_1), \ldots, h^*(t_n))$

Se t é um termo, o elemento $h^*(t) \in V$ será chamado **a interpretação de t por h^***.

Demonstração:
Prova por indução estrutural, sobre a construção do termo t, da propriedade $P(t)$: "existe um único $y = h^*(t)$ verificando (B_h) e (I_h)".
$\diamond$

Acabamos aqui de definir formalmente a noção de interpretação de termos, ou seja de definir a capacidade da linguagem (de termos) de representar algo; Por exemplo que o termo *zero* pode representar o inteiro 0.

Exemplo 4.2 *Seja $F_0 = \{a\}$ e $F_1 = \{s\}$, $F = F_0 \cup F_1$. Temos então $T = \{a, s(a), s(s(a)), \ldots\}$. Suponhamos que $V = \mathbb{N}$.*

1. *se $h_1(a) = 0$ e $h_{1s}(n) = n + 1$, então*

$$h_1^*(s^n(a)) = h_1^*(\underbrace{s(s\ldots}_{n \text{ vezes}}(a)\ldots)) = n$$

2. *se $h_2(a) = 1$ e $h_{2s}(n) = 2n$, então*

$$h_2^*(s^n(a)) = h_2^*(\underbrace{s(s\ldots}_{n \text{ vezes}}(a)\ldots)) = 2^n$$

3. se $h_3(a) = 1$ *e* $h_{3s}(n) = n + 2$*, então*

$$h_3^*(s^n(a)) = h_3^*(\underbrace{s(s\ldots}_{n\ vezes}(a)\ldots)) = 2n + 1$$

De facto verifica-se por exemplo para o primeiro exemplo que $h_1^*(a) = h_1(a) = 0$ *e que* $h_1^*(s^{n+1}(a)) = h_1^*(s(s^n(a))) = h_1^*(s^n(a)) + 1 = n + 1$.

Seja E um conjunto qualquer, e seja X a parte de E definida indutivamente pelas condições (B) e (I). O teorema 3.1 afirma que cada elemento de X é obtido a partir da base aplicando um número finito de passos indutivos. Vamos agora realçar este resultado descrevendo por um termo a forma como um elemento x deste conjunto é obtido.

A cada elemento b da base B, associa-se um símbolo notado $\bar{b}$ de cardinalidade 0. A cada função ϕ de K associa-se o símbolo $\bar{\phi}$ de aridade $\sharp(\phi)$. Seja T o conjunto de todos os termos construídos com esses símbolos.

Considera-se a interpretação $h^* : T \to E$ definida por

- $h(\bar{b}) = b$
- $h_{\bar{\phi}}(x_1, \ldots, x_{\sharp(\phi)}) = \phi(x_1, \ldots, x_{\sharp(\phi)})$

A proposição seguinte confirma o estatuto indutivo de X, ou seja, que X coincide com o conjunto das interpretações dos termos de T.

Proposição 4.2 (Termos e construção indutiva)

$$X = \{h^*(t)/t \in T\}$$

Demonstração:
Basta provar quase trivialmente as propriedades:
$P(x)$: ”existe um termo t tal que $x = h^*(t)$.
$Q(t)$: ”$h^*(t) \in X$”.
$\diamond$

4.3. Não ambiguidade

Definição 4.2 (Definição não ambígua) *Uma definição indutiva dum conjunto X é qualificada de não ambígua se a aplicação h^* da proposição 4.2, além de ser sobrejectiva* [9]*, é injectiva, isto é, se para todo o $x \in X$ existe um e um só termo t tal que $x = h^*(t)$.*

Intuitivamente a não ambiguidade exprime o facto de existir uma e uma só forma de construir o elemento x de X.

Exemplo 4.3 *A definição seguinte de $\mathbb{N}^2$ é ambígua:*

(B) $(0,0) \in \mathbb{N}^2$

(I_1) $(n,m) \in \mathbb{N}^2 \Longrightarrow (n+1,m) \in \mathbb{N}^2$

(I_2) $(n,m) \in \mathbb{N}^2 \Longrightarrow (n,m+1) \in \mathbb{N}^2$

De facto, o par $(1,1)$ obtem-se a partir da base usando a sequência de regras $(B),(I_1),(I_2)$ ou alternativamente $(B),(I_2),(I_1)$. Formalmente considera-se os termos formados a partir:

- *do símbolo $\bar{b}$ de cardinalidade 0 e cuja a interpretação $h(\bar{b})$ é $(0,0)$;*
- *dos símbolos unários $\bar{f}$ e $\bar{g}$ cujas interpretações são definidas por*
  - $h_{\bar{f}}(n,m) = (n+1,m)$
  - $h_{\bar{g}}(n,m) = (n,m+1)$

Temos efectivamente $(1,1) = h^*(\bar{f}(\bar{g}(\bar{b}))) = h^*(\bar{g}(\bar{f}(\bar{b})))$.

[9] O h^* da proposição 4.2 é por definição sobrejectiva visto que para todo o $x \in X$, existe um $t \in T$ tal que $x = h^*(t)$.

5. Funções definidas por recursividade estrutural

5.1. Conceitos e aplicações

Definição 5.1 (Funções definidas por recursividade estrutural) *Seja $X \subseteq E$ um conjunto definido indutivamente de forma não ambígua. Seja F um conjunto qualquer. A definição de uma aplicação ψ de X para F consiste:*

(B_ψ) no dado de $\psi(x)(\in F)$, para todos os $x \in B$,

(I_ψ) na expressão de $\psi(\phi(x_1, \ldots, x_{\sharp(\phi)}))$ a partir dos $x_1, \ldots, x_{\sharp(\phi)}$ e dos $\psi(x_1), \ldots, \psi(x_{\sharp(\phi)})$ para todos os $\phi \in K$. Escrever-se-á

$$\psi(\phi(x_1, \ldots, x_{\sharp(\phi)})) = \psi_\phi(x_1, \ldots, x_{\sharp(\phi)}, \psi(x_1), \ldots, \psi(x_{\sharp(\phi)}))$$

onde ψ_ϕ é uma aplicação de $E^{\sharp(\phi)} \times F^{\sharp(\phi)}$ em F

Exemplo 5.1 *Vejamos alguns exemplos de tais funções recursivas sobre conjuntos indutivos:*

1. *A função factorial definida de $\mathbb{N}$ para $\mathbb{N}$ define-se facilmente por*

   (B) $Fact(0) = 1$

   (I) $Fact(n+1) = (n+1) \times Fact(n)$.

   ou mais naturalmente $Fact(n) = \begin{cases} 1 & se\ n = 0 \\ n \times Fact(n-1) & senão \end{cases}$

2. *A altura de uma árvore x define-se indutivamente por (usando a formulação "natural"):*

$$altura(x) = \begin{cases} 0 & se\ x = \emptyset \\ 1 + max(altura(g), altura(d)) & se\ x = (a, g, d) \end{cases}$$

3. *O percurso infixo de uma árvore (listagem dos nós de uma árvore lendo-os de esquerda para a direita) é definida por:*

$$inf(x) = \begin{cases} \epsilon & se\ x = \emptyset \\ inf(g).a.inf(d) & se\ x = (a, g, d) \end{cases}$$

4. *Seja E o conjunto indutivo das expressões aritméticas (infixas) construidas exclusivamente com as operações $+$ e $\times$ a partir do conjunto de indentificadores A. A função de transformação de uma expressão em notação infixa ($a + b$ por exemplo) para a notação postfixa ($ab+$) define-se por:*

$$post(x) = \begin{cases} a & se\ x = a, a \in A \\ post(e)post(f)+ & se\ x = (e + f) \\ post(e)post(f)\times & se\ x = (e \times f) \end{cases}$$

5. *A condição de não ambiguidade do conjunto indutivo na definição de uma função justifica-se pela razão seguinte: se a definição for ambígua ariscamo-nos a não poder definir funções mas sim relações. Por exemplo retomando a definição ambígua de $\mathbb{N}^2$, a definição*

   (B) $\psi(0, 0) = 1$

   (I_1) $\psi(n + 1, m) = \psi(n, m)^2$

   (I_2) $\psi(n, m + 1) = 3 \times \psi(n, m)$

   não define uma função porque $\psi(1, 1)$ não tem um valor mas dois (9 e 3).

   De forma geral a definição 5.1 define a relação $\mathcal{R}$ entre X e F por

$$x\mathcal{R}y \text{ se e só se } \exists t \text{ termo tal que } x = h^*(t) \text{ e } y = \psi'(t)$$

   onde ψ' é definida como uma função do conjunto dos termos cuja a interpretação é X em F por:

   - $\psi'(\bar{b}) = \psi(b)$

- $\psi'(\bar{\phi}(t_1, \ldots, t_{\sharp(\phi)})) = \psi_\phi(h^*(t_1), \ldots, h^*(t_{\sharp(\phi)}), \psi(t_1), \ldots, \psi'(t_{\sharp(\phi)})$

Se h^* é injectiva esta relação é funcional (isto é, é uma função). Mas não é o único caso. De facto $\mathcal{R}$ é funcional se e só se $\forall t, t' \in T, h^*(t) = h^*(t') \rightarrow \psi'(t) = \psi'(t')$

5.2. Considerações técnicas

É pertinente realçar a importância da propriedade de não ambiguidade a luz do capítulo "Funções Recursivas, Indução Bem Fundada e Pontos Fixos" sobre funções e sistemas recursivos. Considerando o ϕ da definição 5.1, f(x) = ϕ é uma definição válida de f se e só se existe uma única função verificando as equações de ϕ ou, dito de outra forma, se e só se o operador subjacente a ϕ é monótono (ou melhor, contínuo).

A não ambiguidade e as considerações técnicas desenvolvidas na secção 3.4 permitem fundamentar e garantir a existência de tais funções.

Página Pessoal
Página de Rosto
Página 26 de 32
Retroceder
Ecrã Todo
Fechar
Sair

6. Aplicações

ToDo

6.1. Linguagens Lógicas

As linguagens lógicas, que servem de base a lógicas como a lógica proposicional ou ainda a lógica de predicados, constroem as suas frases a volta de várias componentes. As principais construções sintácticas são chamadas *conectores*. Os conectores agrupam símbolos lógicos (proposições ou predicados), símbolos funcionais (funções ou constantes) e variáveis.

Vamos assim considerar nas definições seguintes um conjunto $\mathcal{V}$ numerável (eventualmente infinito) de variáveis (que designaremos de forma preferencial pelas letras x, y e z) e os conjuntos numeráveis de símbolos seguintes:

- o conjunto $\mathcal{P}$ de símbolos de predicado. destacaremos um subconjunto particular de símbolos de predicado, o conjunto $\mathcal{P}^0$ dos símbolos de cardinalidade 0 designados por símbolos proposicionais;
- o conjunto $\mathcal{F}$ de símbolos de função

6.1.1. Linguagem Proposicional

Seja $\mathcal{C}_{prop} \triangleq \{\bot, \top, \wedge, \vee, \neg, \rightarrow, \leftrightarrow\}$ o conjunto de conectores. As cardinalidades dos conectores são as seguintes: $\sharp(\bot) = \sharp(\top) = 0$, $\sharp(\neg) = 1$ e $\sharp(\vee) = \sharp(\wedge) = \sharp(\rightarrow) = \sharp(\leftrightarrow) = 2$.

Definição 6.1 *A linguagem proposicional $\mathcal{P}rop$ é a linguagem de termos definida indutivamente a partir do alfabeto $\mathcal{P}^0 \cup \mathcal{C}_{prop} \cup \{'(',')'\}$ por:*

- *todos os símbolos proposicionais pertencem a $\mathcal{P}rop$;*
- *$\bot$ e $\top$ pertecem a $\mathcal{P}rop$;*
- *se $F_1, F_2 \in \mathcal{P}rop$ então $\neg F_1 \in \mathcal{P}rop$, $(F_1 \wedge F_2) \in \mathcal{P}rop$, $(F_1 \vee F_2) \in \mathcal{P}rop$, $(F_1 \rightarrow F_2) \in \mathcal{P}rop$ e $(F_1 \leftrightarrow F_2) \in \mathcal{P}rop$.*

6.1.2. Linguagem de Predicados

A linguagem de predicados extende a linguagem proposicional ao introduzir conectores chamados quantificadores. Mais do que factos, estes últimos alargam o universo do discurso permitindo expressar relações entre conjuntos de objectos.

A capacidade de quantificar define o grau de expressividade da linguagem em questão. A linguagem que permite quantificar sobre os objectos do discurso é qualificada de primeira ordem. A linguagem que também permite quantificar sobre símbolos que manipulam esses objecto é designada por linguagem de segunda ordem. Uma linguagem de terceira ordem é uma linguagem que permite a quantificação sobre símbolos manipulam símbolos que manipulam objectos, etc ... Designamos por linguagem de predicados de ordem superior a linguagem que permite a quantificação sobre qualquer símbolo da linguagem.

Abordaremos nos próximos capítulos algumas linguagens de ordem superior, vamos aqui concentrar-nos nas linguagens de primeira ordem.

Assim, para poder definir esta linguagem de predicados é preciso, previamente, definir uma linguagem de termos definindo os objectos sobre os quais a linguagem de predicados permitirá discursar.

Definição 6.2 (Linguagem de termos) *O conjunto dos termos $\mathcal{T}$ é o conjunto indutivo definido a partir do alfabeto $\mathcal{F} \cup \mathcal{V} \cup \{'(', ',' ')'\}$ por:*

- *todos os símbolos de constante (símbolos de função de cardinalidade 0) de $\mathcal{F}$ pertencem a $\mathcal{T}$;*
- *todos as variáveis de $\mathcal{V}$ pertencem a $\mathcal{T}$;*
- *se f é um símbolo funcional de cardinalidade $n \geq 1$ e $t_1 \ldots t_n$ termos de $\mathcal{T}$ então $f(t_1, \ldots, t_n) \in \mathcal{T}$*

Definição 6.3 (Fórmulas Atómicas) *O conjunto das fórmulas atómicas é o conjunto das fórmulas da forma $R(t_1, \ldots, t_n)$ onde $R \in \mathcal{P}$ e $\sharp(R) = n$ e $t_1, \ldots, t_n$ termos.*

Seja $\mathcal{C}_{pred} \triangleq \mathcal{C}_{prop} \cup \{\forall, \exists\}$ o conjunto de conectores para a linguagem de predicados.

Página Pessoal
Página de Rosto

Retroceder
Ecrã Todo
Fechar
Sair

Definição 6.4 *A linguagem de predicados de primeira ordem $\mathcal{P}red$ é a linguagem definida indutivamente a partir do alfabeto $\mathcal{P} \cup \mathcal{F} \cup \mathcal{V} \cup \mathcal{C}_{pred} \cup \{'(', ')'\}$ por:*

- *todas as fórmulas atómicas pertencem a $\mathcal{P}red$;*
- *$\perp$ e $\top$ pertecem a $\mathcal{P}rop$;*
- *se $F_1, F_2 \in \mathcal{P}red$ então $\neg F_1 \in \mathcal{P}red$, $(F_1 \wedge F_2) \in \mathcal{P}red$, $(F_1 \vee F_2) \in \mathcal{P}red$, $(F_1 \rightarrow F_2) \in \mathcal{P}red$ e $(F_1 \leftrightarrow F_2) \in \mathcal{P}red$*
- *se x é uma variável e $F \in \mathcal{P}red$ então $\forall v F$ e $\exists v F$ pertencem a $\mathcal{P}red$.*

Dois tipos de variáveis ocorrem numa fórmula F, a variáveis livres e as variáveis ligadas:

Definição 6.5 (Variáveis livres, ligadas e fórmulas fechadas)

- *Uma variável x é* livre *numa fórmula F se nenhuma ocorrência da variável se encontra num sub-termo de F começando por $\forall x$ ou $\exists x$;*
- *no caso contrário diz-se que a variável está* ligada*;*
- *uma fórmula é* fechada *se não tiver nenhuma variável livre.*

Uma operação essencial a linguagens de ordem maior ou igual a 1, que estudaremos mais em detalhe no enquadramento do cálculo λ, é a operação de substituição.

Definição 6.6 (Substituição) *A* substituição *numa fórmula F de uma variável x por um termo t, notada $F[x := t]$, é a operação que consiste em substituir na fórmula F todas ocorrência livres de x o termo t.*

Por exemplo a fórmula obtida pela substituição $((\forall x P(x,x)) \wedge (P(x,y) \rightarrow R(x)))[x := f(z)]$ é a fórmula $((\forall x P(x,x)) \wedge (P(f(z),y) \rightarrow R(f(z))))$.

6.2. Dedução Natural

Vamos nesta secção vamos dar um exemplo de sistema dedutivo definido por indução. Trata-se duma teoria possível para a dedução em lógica de primeira ordem estabelecida pelo lógico G. Gentzen. Todas as considerações que tivemos a propósito nas notações alternativas das definições indutivas tomam aqui um particular valor. A dedução natural trata directamente de estabelecer o que são teoremas em lógica de primeira ordem .

A dedução natural é um formalismo dedicado a construção de provas de fórmulas lógicas.

Definição 6.7 (Juízo) *Um juízo é um par* (Γ, F) *em que* Γ *é um conjunto de fórmulas e* F *uma fórmula. Utilizaremos a notação* $\Gamma \vdash F$*.*

Podemos interpretar um juízo $\Gamma \vdash F$ como uma afirmação: Admitindo as fórmulas de Γ deduz-se $(\vdash)$ F. Utilizar a dedução natural é validar ou desconfirmar tal afirmação.

Definição 6.8 *A dedução natural para a lógica de primeira ordem é o conjunto de regras apresentadas na figura 2.*

Voltamos assim a noção de teorema e de demonstração que destacamos em secções anteriores (ver secção 3.4.0.6)

Definição 6.9 (Teorema e demonstração) $\Gamma \vdash F$ *é designado por* teorema *se pode ser produzida aplicando um número finito de vezes as regras da dedução natural. A árvore de dedução produzida (da qual* $\Gamma \vdash F$ *é a raíz) é designada por* demonstração *de* F *sob as hipóteses de* Γ*.*

Por exemplo o juízo $\vdash (A \to B) \to ((B \to C) \to (A \to C))$ é um teorema e tem a demonstração em dedução natural apresentada na figura 3.

$$axioma\,\frac{A \in \Gamma}{\Gamma \vdash A} \qquad elim_{\perp}\,\frac{\Gamma \vdash \perp}{\Gamma \vdash A}$$

$$intro_{\rightarrow}\,\frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B} \qquad elim_{\rightarrow}\,\frac{\Gamma \vdash A \qquad \Gamma \vdash A \rightarrow B}{\Gamma \vdash B}$$

$$intro_{\wedge}\,\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \qquad elim_{\wedge 1}\,\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}$$

$$elim_{\wedge 2}\,\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B} \qquad intro_{\vee 1}\,\frac{\Gamma \vdash A}{\Gamma \vdash A \vee B}$$

$$intro_{\vee 2}\,\frac{\Gamma \vdash B}{\Gamma \vdash A \vee B} \qquad elim_{\vee}\,\frac{\Gamma \vdash A \vee B \qquad \Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma \vdash C}$$

$$intro_{\neg}\,\frac{\Gamma, A \vdash \perp}{\Gamma \vdash \neg A} \qquad elim_{\neg}\,\frac{\Gamma \vdash A \qquad \Gamma \vdash \neg A}{\Gamma \vdash \perp}$$

$$\text{Dupla Negação}\,\frac{\Gamma \vdash \neg\neg A}{\Gamma \vdash A} \qquad intro_{\leftrightarrow}\,\frac{\Gamma, A \vdash B \qquad \Gamma, B \vdash A}{\Gamma \vdash A \leftrightarrow B}$$

$$elim_{\leftrightarrow 1}\,\frac{\Gamma \vdash A \qquad \Gamma \vdash A \leftrightarrow B}{\Gamma \vdash B} \qquad elim_{\leftrightarrow 1}\,\frac{\Gamma \vdash B \qquad \Gamma \vdash A \leftrightarrow B}{\Gamma \vdash A}$$

$$intro_{\exists}\,\frac{\Gamma \vdash F[x := t] \qquad t\ termo}{\Gamma \vdash \exists x F} \qquad elim_{\exists}\,\frac{\Gamma \vdash \exists x F \qquad \Gamma, F \vdash G}{\Gamma \vdash G}$$

$$intro_{\forall}\,\frac{\Gamma \vdash F \qquad x\ n\tilde{a}o\ livre\ em\ \Gamma}{\Gamma \vdash \forall x F} \qquad elim_{\forall}\,\frac{\Gamma \vdash \forall x F \qquad \forall t\ termo}{\Gamma \vdash F[x := t]}$$

$$
\dfrac{\dfrac{\dfrac{\dfrac{axioma\ \dfrac{}{(A \to B) \vdash (A \to B)} \qquad axioma\ \dfrac{}{A \vdash A}}{(A \to B), A \vdash B} \qquad axioma\ \dfrac{}{(B \to C) \vdash (B \to C)}}{(A \to B), (B \to C), A \vdash C}\ elim_{\to}}{(A \to B), (B \to C) \vdash (A \to C)}\ \to}{\dfrac{(A \to B) \vdash (B \to C) \to (A \to C)}{\vdash (A \to B) \to ((B \to C) \to (A \to C))}}
$$

References

[1] Peter Aczel. An introduction to inductive definitions. In J. Barwise, editor, *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic and the Foundations of Mathematics*, chapter C.7, pages 739–782. North-Holland, Amsterdam, 1977.

[2] A. Arnold and I. Guessarian. *Mathematics for Computer Science*. Prentice-Hall, 1996.

[3] T. Coquand and P. Dybjer. Inductive definitions and type theory: an introduction. In *Foundations of Software Technology and Theoretical Computer Science*, pages 60–76, 1994. `http://www.math.chalmers.se/~peterd/papers/ESSLLI94.ps.gz`

[4] P. Fejer and D. Simovici. *Mathematical Foundations of Computer Science, Volume 1: Sets, Relations and Induction,*. Texts and Monographs in Computer Science. Springer Verlag, 1991.

[5] D. van Dalen. *Logic and Structure*. Springer Verlag, Berlin, Germany, 1983.

[6] G. Winskel. *The Formal Semantics of Programming Languages: An Introduction*. Foundations of Computing series. MIT Press, Cambridge, Massachusetts, February 1993.