5. Programas Universais

5.1 Funções universais e programas universais

5.2 Duas aplicações de programa universal

5.3 Operações efectivas sobre funções computáveis

5.1 Funções Universais e Programas Universais

Considere-se a seguinte função: $y\ (x, y) = f_{x}(y)$

O que faz ?

Tudo o que qualquer função unária computável pode fazer !

Esta função engloba todas as funções unárias computáveis.

Seja **g(y)** uma função qualquer computável. Então, tem um índice **n**.

$$g(y) = f_{n}(y) = y\ (n, y)$$

Diz-se que y é a **função universal** para as funções unárias computáveis.

5.1 Funções Universais e Programas Universais

Definição

A função universal para funções n-árias computáveis é a função (n+1)-ária $y_U^{(n)}$ definida por:

$$y_U^{(n)}(e, x_1, x_2, \ldots, x_n) \approx f_e(x_1, x_2, \ldots, x_n)$$

$y_U^{(n)}$ é computável ?

Se for, existirá um programa $P_U^{(n)}$ capaz de computar qualquer função n-ária computável.

Então, esse programa irá englobar todos os programas e será designado por

Programa Universal.

5.1 Funções Universais e Programas Universais

Teorema

Para cada n, a função universal $y_U^{(n)}$ é computável.

Prova 1: (Recurso à Tese de Church)

Para um **n** fixo e dados um índice **e** e um n-uplo **x**, vamos estabelecer um procedimento informal para computar $y_U^{(n)}$:

> *" Descodificar o número* **e** *e escrever o programa* $\mathbf{P_e}$.
> *Imitar a computação* $\mathbf{P_e(x)}$ *passo por passo.*
> *Se e quando a computação terminar, o valor requerido de* $y_U^{(n)}(e,x)$ *é o número que se encontra em* $\mathbf{R_1}$*."*

Recorrendo à Tese de Church, podemos concluir que $y_U^{(n)}$ é computável.

5.1 Funções Universais e Programas Universais

Prova 2: (Esboço de uma prova formal)

Esquema da prova:

1. usar um único número s para codificar a situação corrente de uma computação

2. Definir uma função computável que exprima a dependência de s relativamente:

(i) ao número do programa: **e**;

(ii) ao vector **x** (input);

(iii) ao número de passos da computação que já foram efectuados: **t**.

5.1 Funções Universais e Programas Universais

- Para **n** fixo, consideremos o índice **e** e o vector **x**. Queremos computar $y_U^{(n)}(e,x)$
- Descodificamos **e** e escrevemos o programa $\mathbf{P_e}$.
- Para um dado **x**, a situação corrente durante a computação $\mathbf{P_e(x)}$ é completamente descrita por:

  - a configuração corrente dos registos $\mathbf{r_1}$, $\mathbf{r_2}$, $\mathbf{r_3}$,…

  - o número **j** da instrução seguinte.

Como apenas um número finito de $\mathbf{r_i}$ é diferente de zero, a configuração corrente pode ser univocamente dada pelo número:

$$c = 2^{r_1}\, 3^{r_2}\, 5^{r_3} \ldots = \prod_{i \geq 1} p_i^{r_i}$$

código de configuração

i-ésimo número primo

Note-se que o conteúdo de $\mathbf{R_i}$ pode ser recuperado a partir de **c**, pois $\mathbf{r_i = (c)_i}$.

5.1 Funções Universais e Programas Universais

A descrição da situação corrente pode assim ser completa através :

$$s = p\,(\mathrm{c},\ \mathrm{j})$$

estado corrente da computação Pe(x)

Convenção*: Se a computação parou, então* ***j = 0*** *e a configuração final é* ***c****.*

Mas os valores de **c**, **j** e s variam durante a computação:

Vamos exprimir a sua dependência relativamente
- ao número do programa (**e**)
- ao input (**x**)
- ao número de passos já efectuados (**t**)

através das seguintes funções (n+2)-árias.

5.1 Funções Universais e Programas Universais

$\mathbf{c_n(e, x, t)}$ = configuração após **t** passos de $\mathbf{P_e(x)}$

(= configuração final, se **Pe(x)** parou em **t** ou menos passos)

$$\mathbf{j_n(e, x, t)} = \begin{cases} \text{número da instrução seguinte de } \mathbf{P_e(x)} \text{ quando } \mathbf{t} \text{ passos foram executados} & \text{, se } \mathbf{P_e(x)} \text{ não parou após } \mathbf{t} \text{ ou menos passos} \\ 0 & \text{, se } \mathbf{P_e(x)} \text{ parou após } \mathbf{t} \text{ ou menos passos} \end{cases}$$

s_n **(e, x, t)** = estado da computação $\mathbf{P_e(x)}$ após **t** passos

$= p\,(c_n(e, x, t), j_n(e, x, t))$

5.1 Funções Universais e Programas Universais

O nosso objectivo agora será provar que σ_n .
Vamos supor que é computável. (provaremos depois)

Então, se a computação **P_e(x)** termina,

o número de passos será $\mu t\,(\varphi_n(e,\ x,\ t) = 0)$

a configuração final será $c_n\,(e,\ x,\ \mu t\,(\varphi_n(e,\ x,\ t) = 0))$

e o resultado final (o conteúdo do registo **R_1**) será:

$$(c_n\,(e,\ x,\ \mu t\,(\varphi_n(e,\ x,\ t) = 0)))_1 = \psi_U^{(n)}(e, x)$$

e, então $\psi_U^{(n)}(e, x)$ é computável. **(c.q.d.)**

Usamos agora a Tese de Church para provar que σ_n é computável.

Damos o seguinte algoritmo informal para determinar efectivamente

$\sigma_n(e,\ x,\ t+1)$ a partir de $\sigma_n(e,\ x,\ t)$ e do índice **e**:

5.1 Funções Universais e Programas Universais

“ Descodificar o número $s_n(e, x, t)$ e obter os **c = c_n(e, x, t)** e **j = j_n(e, x, t)**.

Se j = 0, $s_n(e, x, t+1) = s_n(e, x, t)$

Senão, escrever a configuração codificada em **c**:

$(c)_1$	**$(c)_2$**	**$(c)_3$**	**…**	**$(c)_m$**	**0**	*

Descodificar o índice **e**, obtendo-se o programa **P_e**.

Determinar a instrução **j** de **P_e** e aplicá-la à configuração (*). dando origem a uma nova configuração, codificada pelo número **c’**.

Encontrar o número j’ da próxima instrução. Temos agora: $s_n(e, x, t+1) = p(c', j')$

Então, $s_n(e, x, t)$ é computável por recursão em **t**

e, para **t = 0**, $s_n(e, x, 0) = p(2^{x_1}3^{x_2}\ldots p_n^{x_n}, 1)$

Recorrendo à Tese de Church, $s_n(e, x, t)$ é e o teorema fica provado.

Uff !!

5.1 Funções Universais e Programas Universais

Corolário

Para cada **n > 0**, os seguintes predicados são decidíveis:

(a) $S_n(e, x, y, t) \equiv$ ' $P_e(x) \downarrow$ y em t ou menos passos'

(b) $H_n(e, x, t) \equiv$ ' $P_e(x) \downarrow$ em t ou menos passos'

Prova: (a) $S_n(e, x, y, t) \equiv$ ' $j_n(e, x, t) = 0$ e $(c_n(e, x, t))_1 = y$ '

(b) $H_n(e, x, t) \equiv$ ' $j_n(e, x, t) = 0$ '

5.1 Funções Universais e Programas Universais

Corolário

Existe uma função total computável **U(x)** e, para cada **n > 0**, um predicado decidível **T_n(e, x, z)**,tais que:

(a) $f_e^{(n)}(x)$ está definida se e só se $\exists z : T_n(e,x,z)$

(b) $f_e^{(n)}(x) \approx U(m z\, T_n(e,x,z))$

Prova: Para determinar onde $f_e^{(n)}(x)$ está definida e o valor que toma, é necessário procurar um par de números **y**, **t**, tais que $S_n(e,x,y,t)$.

O operador m permite-nos perquisar efectivamente por um único número que possua uma determinada propriedade . Para o usar na pesquisa de um par de números, podemos tomar o número **z** como uma codificação dos números **$(z)_1$** e **$(z)_2$**.

Então, definimos: $T_n(e,\ x,\ z) = S_n(e,x,(z)_1,(z)_2)$

5.1 Funções Universais e Programas Universais

Prova (cont.):

Para **(a)**, vamos supor que $f_e^{(n)}(x)$ está definida; então, existem **y**, **t**, tais que $S_n(e, x, y, t)$

Fazendo $z = 2^y\ 3^t$ temos $T_n(e,\ x,\ z)$

Inversamente, se existe um z tal que $T_n(e,\ x,\ z)$,

então, pela definição de **T$_n$**, $P_e(x)\downarrow$ e $f_e^{(n)}(x)$ está definida.

Para **(b)**, é claro da definição de **T$_n$** que, se $f_e^{(n)}(x)$ está definida, então para qualquer **z** tal que $T_n(e,\ x,\ z)$, temos $f_e^{(n)}(x) = (z)_1$.

Fazendo **U(z) = (z)$_1$**, vem $f_e^{(n)}(x) \approx U(\mu z\ T_n(e, x, z))$

□

5.2 Uma aplicação do Programa Universal

O problema “ f_x é total” é indecidível

Seja **g** a função característica deste problema, ou seja: $$\mathrm{g(x)} = \begin{cases} 1 \ , \text{ se } f_{\mathrm{x}} \text{ e total} \\ 0 \ , \text{ se } f_{\mathrm{x}} \text{ nao e total} \end{cases}$$

Vamos mostrar que **g** não é computável.

*(Por redução ao absurdo, usamos o método da diagonal para construir uma função total **f** diferente de todas as funções computáveis, por forma que, se **g** é computável, também o seria **f**.)*

Seja **f** definida por: $$\mathrm{f(x)} = \begin{cases} f_{\mathrm{x}}(x)+1 \ , \text{ se } f_{\mathrm{x}} \text{ e total} \\ \quad 0 \quad\quad\ , \text{ se } f_{\mathrm{x}} \text{ nao e total} \end{cases}$$

Claramente, **f** é total e não computável. Usando **g** e y_U , podemos escrever f como:

$$\mathrm{f(x)} = \begin{cases} y_{\mathrm{U}}(x,x)+1 \ , \text{ se } g(\mathrm{x}) = 1 \\ \quad 0 \quad\quad\quad\ , \text{ se } g(\mathrm{x}) = 0 \end{cases}$$

Se **g** fosse computável, então como y_U é computável, pela Tese de Church poderíamos concluir que **f** era computável (!!). Logo, **g** não é computável .