1 Funções parciais recursivas

De um modo sucinto pode-se dizer que a classe das funções parciais recursivas é constituída por um certo número de funções atómicas e por funções que se obtêm de outras funções parciais recursivas por composição (substituição), recursão (recorrência) ou minimização. Assim, para definir rigorosamente a classe das funções parciais recursivas, há que definir, em primeiro lugar, quem são as funções atómicas e o que se entende por composição, recursão e minimização.

Notação: Para cada $n \in I\!N_0$,

$$[I\!N_0^n \to I\!N_0]$$

representa a classe das funções que têm conjunto de partida $I\!N_0^n$ e conjunto de chegada $I\!N_0$. O caso em que $n = 0$ corresponde às constantes naturais. As funções em $[I\!N_0^n \to I\!N_0]$ são designadas funções n-árias ou de aridade n.

Definição[1]: FUNÇÕES ATÓMICAS/PRIMITIVAS
Designam-se funções atómicas (ou primitivas) as seguintes funções

- a constante 0, usualmente designada Z (zero)
- $\lambda x.x + 1 \in [I\!N_0 \to I\!N_0]$, usualmente designada S (sucessor)
- para cada $n \in I\!N$ e $1 \leq i \leq n$,

  $$\lambda x_1 \ldots x_n.x_i \in [I\!N_0^n \to I\!N_0]$$

  usualmente designada $U_{n,i}$ (projecção n, i).

Definição: FUNÇÃO DEFINIDA POR COMPOSIÇÃO (OU SUBSTITUIÇÃO)
Sejam $k \in I\!N$, $n \in I\!N_0$, $f \in [I\!N_0^k \to I\!N_0]$ e $g_1, \ldots, g_k \in [I\!N_0^n \to I\!N_0]$. A função em $[I\!N_0^n \to I\!N_0]$

$$h = \lambda x_1 \ldots x_n.f(g_1(x_1, \ldots, x_n), \ldots, g_k(x_1, \ldots, x_n))$$

diz-se definida por composição (ou substituição) a partir de f e $g_1, \ldots, g_k$.

Exemplo:

1. A função $h = \lambda xy.\,|x - y|$ é definida por composição a partir de[2] $f = \lambda xy.x + y$, $g_1 = \lambda xy.x \dot{-} y$ e $g_2 = \lambda xy.y \dot{-} x$, isto é

   $$h(x,y) = f(g_1(x,y), g_2(x,y)) = (x \dot{-} y) + (y \dot{-} x)$$

[1] Estas notas são baseadas em Sintaxe e Semântica de Linguagens I, J-F Costa, DMIST,2000; *Introdução à teoria da computação*, C. Sernadas, Editorial Presença, 1993; *Computability-an introduction to recursive function theory*, N. Cutland, Cambridge University Press, 1980

[2] $x \dot{-} y$ é $x - y$ se $x > y$ e 0 caso contrário

2. A função $h = \lambda xy.max(x, y)$ é definida por composição a partir de $f = \lambda xy.x + y$, $U_{2,2}$ e de $g = \lambda xy.x \dot{-} y$, isto é

$$h(x, y) = f(U_{2,2}(x, y), g(x, y)) = y + (x \dot{-} y)$$

Proposição:
Sejam $n \in \mathbb{N}_0$, $f \in [\mathbb{N}_0^n \to \mathbb{N}_0]$ e $g \in [\mathbb{N}_0^{n+2} \to \mathbb{N}_0]$. Existe uma e uma só função $h \in [\mathbb{N}_0^{n+1} \to \mathbb{N}_0]$ tal que

- $h(x_1, \ldots, x_n, 0) = f(x_1, \ldots, x_n)$
- $h(x_1, \ldots, x_n, y + 1) = g(x_1, \ldots, x_n, y, h(x_1, \ldots, x_n, y))$.

Definição: Função definida por recursão (ou recorrência)
Sejam $n \in \mathbb{N}_0$, $f \in [\mathbb{N}_0^n \to \mathbb{N}_0]$ e $g \in [\mathbb{N}_0^{n+2} \to \mathbb{N}_0]$. A função $h \in [\mathbb{N}_0^{n+1} \to \mathbb{N}_0]$ tal que

- $h(x_1, \ldots, x_n, 0) = f(x_1, \ldots, x_n)$
- $h(x_1, \ldots, x_n, y + 1) = g(x_1, \ldots, x_n, y, h(x_1, \ldots, x_n, y))$.

diz-se definida por recursão (ou recorrência) a partir de f e g.

Observação: É possível definir outras formas de recursão. A definição de recursão acima apresentada é usualmente designada *recursão primitiva*.

Exemplo:

1. A função $h = \lambda xy.x + y$ é definida por recursão a partir da função unária f e da função ternária g seguintes
   - função f:
     - $h(x, 0) = f(x)$
     - $h(x, 0) = x$

     e portanto $f = \lambda x.x$
   - função g:
     - $h(x, k + 1) = g(x, k, h(x, k))$
     - $h(x, k + 1) = x + (k + 1) = (x + k) + 1 = h(x, k) + 1$

     e portanto $g = \lambda xyz.z + 1$
2. A função[3] $h = \lambda x.x \dot{-} 1$ é definida por recursão a partir da constante f e da função binária g seguintes
   - constante f:

[3] $x \dot{-} 1$ é $x - 1$ se $x > 1$ e 0 caso contrário

- $h(0) = f$
- $h(0) = 0$

e portanto $f = 0$

- função g:
  - $h(k+1) = g(k, h(k))$
  - $h(k+1) = (k+1) \dot{-} 1 =^4 (k+1) - 1 = k$

  e portanto $g = \lambda xy.x$

3. A função[5] $h = \lambda xy.x \dot{-} y$ é definida por recursão a partir da função unária f e da função ternária g seguintes

   - função f:
     - $h(x, 0) = f(x)$
     - $h(x, 0) = x$

     e portanto $f = \lambda x.x$
   - função g:
     - $h(x, k+1) = g(x, k, h(x, k))$
     - $h(x, k+1) = x \dot{-} (k+1) = (x \dot{-} k) \dot{-} 1 = h(x, k) \dot{-} 1$

     e portanto $g = \lambda xyz.z \dot{-} 1$

Definição: Operador μ
Sejam $n \in I\!N_0$ e $f \in [I\!N_0^{n+1} \to I\!N_0]$. Para cada $x_1, \ldots, x_n \in I\!N_0$, define-se

$$\mu y.(f(x_1, \ldots, x_n, y) = 0)$$

como o mais pequeno y, se existir, tal que

$$f(x_1, \ldots, x_n, z) \text{ está definida para cada } z \leq y$$
$$\text{e}$$
$$f(x_1, \ldots, x_n, y) = 0.$$

Se não existir um tal y, então o valor de $\mu y.(f(x_1, \ldots, x_n, y) = 0)$ não está definido.

Definição:Função definida por minimização
Sejam $n \in I\!N_0$ e $f \in [I\!N_0^{n+1} \to I\!N_0]$. A função $g \in [I\!N_0^n \to I\!N_0]$ tal que

$$g = \lambda x_1 \ldots x_n.\mu y.(f(x_1, \ldots, x_n, y) = 0)$$

diz-se definida por minimização a partir de f.

Exemplo:

[4] dado que $k + 1 \geq 1$,
[5] $x \dot{-} y$ é $x - y$ se $x > y$ e 0 caso contrário

1. A função $h = \lambda x.$ "$x/4$ se x é múltiplo de 4 e não definida caso contrário" é definida por minimização a partir da função binária

$$f(x,y) = \mid 4y - x \mid$$

isto é

$$h(x) = \mu y(\mid 4y - x \mid = 0)$$

2. A função $h = \lambda x.$ "quociente da divisão inteira de x por 4" é definida por minimização a partir da função binária

$$f(x,y) = (x+1) \dot{-} 4(y+1)$$

isto é

$$h(x) = \mu y((x+1) \dot{-} 4(y+1) = 0)$$

pois $h(x) =$"menor y tal que $x < 4(y+1)$"

3. A função $h = \lambda x.$ "raiz cúbica inteira de x" é definida por minimização a partir da função binária

$$f(x,y) = (x+1) \dot{-} (y+1)^3$$

isto é

$$h(x) = \mu y((x+1) \dot{-} (y+1)^3 = 0)$$

pois $h(x) =$"menor y tal que $x < (y+1)^3$"

Podem agora definir-se a noção de função (parcial) recursiva e outras noções associadas.

Definição:Funções (parciais) recursivas
A classe R das funções (parciais) recursivas[6] é a mais pequena classe que contém as funções atómicas e é fechada para a substituição, recorrência e minimização.

Outra forma, equivalente, de definir esta classe é a seguinte. A classe R é a classe de funções definida indutivamente como se segue:

- todas as funções atómicas pertencem a R
- se $k, n \in \mathbb{N}_0$, f é uma função de aridade k pertencente a R e $g_1, \ldots, g_k$ são funções de aridade n pertencentes a R então a função h definida por composição (ou substituição) a partir de f e $g_1, \ldots, g_k$ também pertence a R
- se $n \in \mathbb{N}_0$, f é uma função de aridade n pertencente a R e g é uma função de aridade $n+2$ pertencente a R então a função h definida por recursão a partir de f e g também pertence a R

[6]muitas vezes, para enfatizar o facto de se tratar aqui de funções (e não necessariamente apenas aplicações) a classe R designa-se *classe das funções parciais recursivas*

- se $n \in \mathbb{N}_0$ e f é uma função de aridade $n+1$ pertencente a R então a função g definida por minimização a partir de f também pertence a R.

Definição: Funções primitivamente recursivas
A classe P_R das funções primitivamente recursivas é a mais pequena classe que contém as funções atómicas e é fechada para a substituição e recorrência.

Um predicado primitivamente recursivo é um predicado cuja característica é primitivamente recursiva.

Como provar que uma função f é parcial recursiva, ou seja, que pertence a R?

Para provar que uma função f é parcial recursiva é necessário mostrar que ela pode ser obtida a partir de funções atómicas e das operações de composição e/ou recursão e/ou minimização. Uma forma de o conseguir pode ser a exibição de uma derivação (ou demonstração) da função, ou seja, uma sequência

$$\begin{array}{c} f_1 \\ f_2 \\ \ldots \\ f_j \\ \ldots \\ f_n \end{array}$$

tal que

- f_n é f
- para cada $1 \leq j \leq n$, f_j é uma função atómica ou é uma função que é obtida por composição, recursão ou minimização a partir de funções em $\{f_1, \ldots, f_{j-1}\}$.

Definição: Derivação de função
Seja $f \in [\mathbb{N}_0^n \to \mathbb{N}_0]$, com $n \in \mathbb{N}_0$. Uma derivação (ou demonstração) de f é uma sequência $f_1 f_2 \ldots f_n$ de funções parciais recursivas tal que f_n é f e, para cada $1 \leq j \leq n$, f_j é uma função atómica ou é uma função que é obtida por composição, recursão ou minimização a partir de funções em $\{f_1, \ldots, f_{j-1}\}$.

Proposição:
Sendo $f \in [\mathbb{N}_0^n \to \mathbb{N}_0]$, com $n \in \mathbb{N}_0$, f é uma função parcial recursiva sse existe uma derivação (ou demonstração) de f.

Exemplo: Apresentam-se seguidamente algumas derivações. Em algumas situações, e para simplificar, pode omitir-se a (sub)derivação de uma dada função indicando que foi provado que essa função é parcial recursiva (abreviadadamente,"f.p.r.").

- A função $h = \lambda xy.x + y$ é parcial recursiva:

1. $\lambda x.x$ — $U_{1,1}$
2. $\lambda xyz.z$ — $U_{3,3}$
3. $\lambda x.x+1$ — S
4. $\lambda xyz.z+1$ — $C(3;2)$
5. $\lambda xy.x+y$ — $R(1,4)$

Uma outra forma de mostrar que $h = \lambda xy.x+y$ é parcial recursiva consiste em construir uma expressão que ilustre o modo como a h é obtida a partir das funções atómicas e das operações de composição, recursão e minimização: $h = R(U_{1,1}, C(S; U_{3,3}))$.

- A função $h = \lambda x.x \dot{-} 1$ é parcial recursiva:

1. 0 — Z
2. $\lambda xy.x$ — $U_{2,1}$
3. $\lambda x.x \dot{-} 1$ — $R(1,2)$

Uma outra forma de mostrar que $h = \lambda x.x \dot{-} 1$ é parcial recursiva consiste em construir uma expressão que ilustre o modo como a h é obtida a partir das funções atómicas e das operações de composição, recursão e minimização: $h = R(Z, U_{2,1})$.

- A função $h = \lambda x.xy \dot{-} y$ é parcial recursiva:

1. $\lambda x.x$ — $U_{1,1}$
2. $\lambda xyz.z$ — $U_{3,3}$
3. $\lambda x.x \dot{-} 1$ — f. p. r. (provado)
4. $\lambda xyz.z \dot{-} 1$ — $C(3;2)$
5. $\lambda xy.x \dot{-} y$ — $R(1,4)$

- A função $h = \lambda xy. |x-y|$ é parcial recursiva:

1. $\lambda xy.x \dot{-} y$ — f. p. r. (provado)
2. $\lambda xy.x+y$ — f. p. r. (provado)
3. $\lambda xy.x$ — $U_{2,1}$
4. $\lambda xy.y$ — $U_{2,2}$
5. $\lambda xy.y \dot{-} x$ — $C(1;4,3)$
6. $\lambda xy. |x-y|$ — $C(2;1,5)$

2 Funções parciais recursivas e funções URM-computáveis

Prova-se que a classe das funções parciais recursivas coincide com a classe das funções URM-computáveis, isto é, $R = C$. A prova pressupõe que se mostre, em particular, que as funções atómicas são URM-computáveis e que as funções obtidas por composição, recursão ou minimização a partir de funções URM-computáveis são também URM-computáveis. O facto de $R = C$ permite que se possam fazem *provas axiomáticas de computabilidade.*

Seguidamente apresentam-se as proposições que estabelecem estes resultados.

Proposição:
As funções atómicas são computáveis pela máquina URM.

Prova:

- Z é calculada pelo programa $Z(1)$
- S é calculada pelo programa $S(1)$
- $U_{n,i}$, para cada $n \in I\!N$ e $1 \leq i \leq n$, é calculada pelo programa $T(i, 1)$

Proposição:
Sejam $k, n \in I\!N_0$, $f \in [I\!N_0^k \to I\!N_0]$ e $g_1, \ldots, g_k \in [I\!N_0^n \to I\!N_0]$. Se f e $g_1, \ldots, g_k$ são funções URM-computáveis então a função h definida por substituição (ou composição) a partir de f e $g_1, \ldots, g_k$ também é URM-computável.

Prova: Sendo F, $G_1, \ldots, G_k$ programas normalizados para f e $g_1, \ldots, g_k$, respectivamente, e

$$m = max\{n, k, \rho(F), \rho(G_1), \ldots, \rho(G_n)\}$$

o programa H que calcula h é

$$\begin{array}{l} T(1, m+1) \\ \ldots \\ T(n, m+n) \\ G_1[m+1, \ldots, m+n \to m+n+1] \\ \ldots \\ G_k[m+1, \ldots, m+n \to m+n+k] \\ F[m+n+1, \ldots, m+n+k \to 1] \end{array}$$

Proposição:
Sejam $n \in I\!N_0$, $f \in [I\!N_0^n \to I\!N_0]$ e $g \in [I\!N_0^{n+2} \to I\!N_0]$. Se f e g são funções URM-computáveis então a função h definida por recursão a partir de f e g também é URM-computável.

Prova: Sendo F e G programas normalizados para f e g, respectivamente, e

$$m = max\{n+2, \rho(F), \rho(G)\}$$

o programa H que calcula h é

$$
\begin{array}{rl}
 & T(1, m+1) \\
 & \ldots \\
 & T(n+1, m+n+1) \\
 & F[1, \ldots, n \to m+n+3] \\
p_{l_2} & J(m+n+2, m+n+1, l_1) \\
 & G[m+1, \ldots, m+n, m+n+2, m+n+3 \to m+n+3] \\
 & S(m+n+2) \\
 & J(1, 1, l_2) \\
p_{l_1} & T(m+n+3, 1)
\end{array}
$$

Proposição:
Seja $n \in I\!N_0$ e $f \in [I\!N_0^{n+1} \to I\!N_0]$ uma função URM-computável. A função g definida a partir de f por minimização é URM-computável.

Prova: Sendo F programa normalizado para f e

$$m = max\{n+1, \rho(F)\}$$

o programa G que calcula g é

$$
\begin{array}{rl}
 & T(1, m+1) \\
 & \ldots \\
 & T(n, m+n) \\
p_{l_2} & F[m+1, \ldots, m+n+1 \to 1] \\
 & J(1, m+n+2, l_1) \\
 & S(m+n+1) \\
 & J(1, 1, l_2) \\
p_{l_1} & T(m+n+1, 1)
\end{array}
$$

Proposição: Tem-se que $R = C$, isto é a classe das funções parciais recursivas coincide com a classe das funções URM-computáveis.

Prova (esboço): Prova-se seguidamente que $R \subseteq C$. A prova do caso $C \subseteq R$ pode ser consultada, por exemplo, num dos livros recomendados na bibliografia da cadeira: *Computability-an introduction to recursive function theory*, N. Cutland, Cambridge University Press, 1980.

Como se viu anteriormente, uma função f é parcial recursiva (ou seja, pertence a R) sse existe uma demonstração, ou derivação, da função, ou seja, uma sequência

$$f_1 f_2 \ldots f_n$$

tal que

- f_n é f
- para cada $1 \leq j \leq n$, f_j é uma função atómica ou é uma função que é obtida por composição, recursão ou minimização a partir de funções em $\{f_1, \ldots, f_{j-1}\}$.

A prova de que $R \subseteq C$ decorre por indução no comprimento das derivações (isto é, no número de elementos da derivação).

Prova da base: Mostra-se que qualquer função $f \in R$ que tenha uma derivação de comprimento 1 é URM-computável (isto é, $f \in C$). De facto, pela definição de derivação, se f tem derivação de comprimento 1, f é o único elemento da derivação e necessariamente f é função atómica. Como todas as funções atómicas são URM-computáveis, o resultado fica estabelecido.

Prova da passo: Mostra-se que qualquer função $f \in R$ que tenha uma derivação de comprimento n, $n > 1$, é URM-computável (isto é, $f \in C$) admitindo, por *hipótese de indução* que funções em R que tenham derivação de comprimento menor que n são URM-computáveis.

Seja $f \in R$ com derivação de comprimento n

$$f_1 f_2 \dots f_n.$$

Sendo $k < n$, derivação $f_1 f_2 \dots f_k$ é uma derivação de f_k e portanto $f_k \in R$. A derivação de f_k comprimento menor que n, logo, pela hipótese de indução, f_k é URM-computável. Conclui-se assim que, para cada $k < n$, f_k é URM-computável.

Por definição de derivação, f_n é uma função atómica ou é uma função que é obtida por composição, recursão ou minimização a partir de funções em $\{f_1, \dots, f_{n-1}\}$. No primeiro caso, como todas as funções atómicas são URM-computáveis, o resultado fica estabelecido. Nos outros casos, o resultado fica também estabelecido porque (i) pelo parágrafo anterior, $f_1, \dots, f_{n-1}$ são URM-computáveis e (ii) pelas proposições anteriores, funções obtidas por composição, recursão ou minimização a partir de funções URM-computáveis são também URM-computáveis.

Observação: Uma outra forma de provar que $R \subseteq C$ consiste em fazer uma prova por indução sobre o conjunto (indutivamente definido) R. Neste caso, a *prova da base* de indução consiste em provar que todas as funções atómicas são URM-computáveis. A *prova do passo* de indução consiste em provar que (i) se $f \in [\mathbb{N}_0^k \to \mathbb{N}_0]$ e $g_1, \dots, g_k \in [\mathbb{N}_0^n \to \mathbb{N}_0]$, $k, n \in \mathbb{N}_0$, são URM-computáveis então a função obtida por composição de f com $g_1, \dots, g_k$ é URM-computável; (ii) se $f \in [\mathbb{N}_0^n \to \mathbb{N}_0]$ e $g \in [\mathbb{N}_0^{n+2} \to \mathbb{N}_0]$, $n \in \mathbb{N}_0$, são URM-computáveis então a função obtida por recursão a partir de f e g é URM-computável e (iii) se $f \in [\mathbb{N}_0^{n+1} \to \mathbb{N}_0]$, $n \in \mathbb{N}_0$, é URM-computável então a função obtida por minimização a partir de f é URM-computável.

3 Provas axiomáticas de computabilidade

O facto de $R = C$ permite provar que uma função é URM-computável sem construir um programa que a calcula. Tais provas designam-se por provas *axiomáticas* de computabilidade.

De facto, como $R = C$, se se provar que uma função f pertence a R, exibindo a correspondente derivação, tem-se também que f pertence a C. Assim, prova-se que uma função é URM-computável não directamente através da construção de um programa que a calcula, mas de uma prova de que a função é parcial recursiva. A derivação referida é a *prova axiomatica da computabilidade de f*.

Proposição:
Se M e Q são predicados n-ários, $n \in I\!N_0$, decidíveis então também são decidíveis os predicados

- $\neg M$
- $M \wedge Q$
- $M \vee Q$

Prova: Tem-se que

- $c_{\neg M}$ é $\lambda x_1 \ldots x_n.1 \dot{-} c_M(x_1, \ldots, x_n)$
- $c_{M \wedge Q}$ é $\lambda x_1 \ldots x_n.c_M(x_1, \ldots, x_n)c_Q(x_1, \ldots, x_n)$
- $c_{M \vee Q}$ é $\lambda x_1 \ldots x_n.max(c_M(x_1, \ldots, x_n), c_Q(x_1, \ldots, x_n))$

Proposição:
Seja $n \in I\!N_0$ e R um predicado $n+1$-ário decidível então a função

$$\lambda x_1, \ldots, x_n. \begin{cases} \text{o menor } y\text{, se existir, tal que } R(x_1, \ldots, x_n, y) \\ \\ \text{indefinida, caso contrário} \end{cases}$$

é URM-computável.

Prova: A função em questão é igual a

$$\lambda x_1, \ldots, x_n.\mu y.(\underline{sg}(c_R(x_1, \ldots, x_n, y)) = 0)$$