08. Modelos com Recursão

Princípios de Engenharia de Software (Texto em Elaboração)

Italo S. Vega
italo@pucsp.br

Faculdade de Estudos Interdisciplinares (FACEI)

Pontifícia Universidade Católica de São Paulo

Sumário

C.A.R. Hoare — Programmers are always surrounded by complexity; we cannot avoid it…. If our basic tool, the language in which we design and code our programs, is also complicated, the language itself becomes part of the problem rather than part of its solution.

8 Modelos com Recursão

 — Consigo **definir** uma função recursiva e **discutir** sobre um modelo com funções recursivas.

8.1 Recursão

A cifragem de uma palavra envolve a cifragem de cada um dos seus símbolos constituintes. Por exemplo, qual a cifra da sequência $w \triangleq \langle t, u, r, i, n, g \rangle = \langle turing \rangle$? Ou, em termos da linguagem Python, qual o resultado da avaliação da chamada `cifraCadeia ("turing")`?“, pensa Fubã. Para ajudá-lo, precisamos de operações adicionais sobre sequências. Entretanto, Fubã resolve seguir por um caminho recursivo para especificar a função $cifraCadeia$.

 — Muito embora não saiba como fazer isso...

Recursão A recursão é uma técnica de programação que se baseia em uma definição realizada sobre si mesma (Lewis & Loftus, 2011, p. 466). Definem-se muitas fórmulas matemáticas recursivamente. Espec considera um exemplo simples para ilustrar a ideia de programação recursiva: a soma dos valores entre 1 e n, inclusive, com $n > 0$.

 — Tal soma pode ser expressa como n mais a soma dos outros $n - 1$ valores:

$$\begin{aligned}\sum_{i=1}^{n} i &= n + \sum_{i=1}^{n-1} i = n + n - 1 + \sum_{i=1}^{n-2} i \\ &= n + n - 1 + n - 2 + \sum_{i=1}^{n-3} i = \cdots \\ &= n + n - 1 + n - 2 + n - 3 + \cdots + 2 + 1\end{aligned}$$

Espec introduz uma forma-λ para especificar a função $soma$ recursivamente:

$$soma = \lambda n \bullet \textbf{if } n = 1 \textbf{ then } 1 \textbf{ else } n + soma(n - 1)$$

Em seguida, ele ilustra o cálculo da soma dos três primeiros números mecanicamente:

$$
\begin{aligned}
soma(3) &= (\lambda n \bullet \textbf{if } n = 1 \textbf{ then } 1 \textbf{ else } n + soma(n-1)) \ (3) \\
&= 3 + soma(3-1) \\
&= 3 + (\lambda n \bullet \textbf{if } n = 1 \textbf{ then } 1 \textbf{ else } n + soma(n-1)) \ (2) \\
&= 3 + 2 + soma(2-1) \\
&= 3 + 2 + (\lambda n \bullet \textbf{if } n = 1 \textbf{ then } 1 \textbf{ else } n + soma(n-1)) \ (1) \\
&= 3 + 2 + 1 = 6
\end{aligned}
$$

 — Consigo implementar, facilmente, esta função usando o método `soma` em Python:

```
# CENÁRIO de implementação de uma função recursiva
def soma (n) :
  if (n == 1) :
    return 1
  return n + soma (n-1)
# uso do método "soma"
soma (3)
```

De repente aparece o Ocara, afirmando conhecer uma maneira mais direta de implementar a função *soma* em Python.

 — Não precisa de recursão! É só usar a função `sum` da biblioteca... bem mais simples!

Ele pega o teclado do Fubã e digita:

```
# CENÁRIO de implementação de uma função recursiva usando "for"
def soma (n) :
  return sum(range(1, n+1))
# uso do método "soma"
soma (3)
```

Espec concorda que o código do Ocara é mais "direto" em relação à versão recursiva. Ele tenta explicar que o problema de cálculo da somatória foi utilizado para demonstrar um programa recursivo simples e não para ser solucionado por recursão. Um programador deverá decidir a respeito de uma implementação apropriada, caso a caso.

 — E como faço a cifragem de uma sequência de símbolos, como $\langle\ turing\ \rangle$, por exemplo?

8.2 Implementação de Recursões

Espec adota a palavra "mel", modelada pela sequência $w \triangleq \langle m, e, l \rangle$, como exemplo no seu raciocínio:

$$cifraCadeia(\langle m, e, l \rangle) = ?$$

 — Podemos cifrar o primeiro símbolo w_1 e concatenar com a cifragem dos símbolos restantes.

Ele utiliza $\frown$ para representar a concatenação entre duas sequências e começa pela cifragem do último símbolo, l. Usando o resultado da cifragem da letra representada pelo símbolo l, Espec constrói uma sequência unitária. A seguir, concatena esta sequência com o resultado da cifragem dos símbolos restantes, aplicando a função $cifraCadeia$ recursivamente:

$$cifraCadeia(\langle m, e, l \rangle) = cifraCadeia(\langle m, e \rangle) \frown \langle cifraSimb(l) \rangle = ?$$

 — Entendi!

$$\begin{aligned} cifraCadeia(\langle m, e, l \rangle) &= cifraCadeia(\langle m, e \rangle) \frown \langle cifraSimb(l) \rangle \\ &= cifraCadeia(\langle m, e \rangle) \frown \langle o \rangle \\ &= cifraCadeia(\langle m \rangle) \frown \langle cifraSimb(e) \rangle \frown \langle o \rangle \\ &= cifraCadeia(\langle m \rangle) \frown \langle h \rangle \frown \langle o \rangle \\ &= \langle cifraSimb(m) \rangle \frown \langle h \rangle \frown \langle o \rangle \\ &= \langle p \rangle \frown \langle h \rangle \frown \langle o \rangle \\ &= \langle p, h, o \rangle \end{aligned}$$

Ocara observa com ar desconfiado e tenta implementar esta computação de forma iterativa. Espec percebe a intenção de Ocara e começa a escrever a definição recursiva da função $cifraCadeia$, começando pelo seu tipo:

$$cifraCadeia \ : \ \mathsf{seq}\ (\mathbf{ran} A) \to \mathsf{seq}\ (\mathbf{ran} A)$$

O domínio é constituído por todas as sequências que podem ser formadas a partir dos símbolos de A. Da mesma forma quanto ao codomínio.

Base de Recursão A função $cifraCadeia$ retorna a entrada cifrada por $cifraSimb$, se houver apenas um único símbolo:

$$cifraCadeia_1 \triangleq \{\ w : \mathsf{seq}\ (\mathbf{ran} A) \mid (\#w = 1) \bullet w \mapsto \langle cifraSimb(w_1) \rangle\ \}$$

Pergunta. *Contexto* Considere um alfabeto modelado pela sequência $t = \langle 4, 5, 6 \rangle$. Definiu-se a função:

$$f_1 \triangleq \{\ w : \mathsf{seq}\ (\mathbf{ran}\ t) \mid (\#w = 1) \bullet w \mapsto \langle w_1 + 3 \rangle\ \}$$

Enunciado Assinale a alternativa contendo uma afirmação verdadeira:

1. $f_1(\langle 2 \rangle) = 5$
2. $f_1(\langle 6 \rangle) = \langle 6 \rangle$
3. $f_1(\langle 3 + 1 \rangle) = 4$
4. $f_1 = \{\langle 4 \rangle \mapsto \langle 7 \rangle, \langle 5 \rangle \mapsto \langle 8 \rangle, \langle 6 \rangle \mapsto \langle 9 \rangle\}$

Passo Recursivo Caso a entrada seja uma sequência $w = s \frown \langle w_n \rangle$ com mais do que um símbolo, aplica-se $cifraSimb$ sobre o último elemento w_n e $cifraCadeia$, recursivamente, sobre os demais símbolos em s:

$$\begin{aligned} cifraCadeia_2 \triangleq \{\ & w : \mathsf{seq}\ (\mathbf{ran} A) \mid (w = s \frown \langle w_n \rangle) \wedge (\#w > 1) \\ & \bullet w \mapsto (cifraCadeia(s) \frown cifraCadeia_1(\langle w_n \rangle))\ \} \end{aligned}$$

 — A definição de $cifraCadeia$ combina estas duas partes:

$$cifraCadeia \triangleq cifraCadeia_1 \cup cifraCadeia_2$$

Pergunta. *Contexto* Considere um alfabeto modelado pela sequência $t = \langle 4, 5, 6 \rangle$. Definiu-se a função:

$$\begin{aligned} f_1 \triangleq & \{\ w : \mathsf{seq}\ (\mathbf{ran}\ t) \mid (\#w = 1) \bullet w \mapsto \langle w_1 + 3 \rangle\ \} \\ f_2 \triangleq & \{\ w : \mathsf{seq}\ (\mathbf{ran}\ t) \mid (w = s \frown \langle w_n \rangle) \wedge (\#w > 1) \\ & \bullet w \mapsto (f(s) \frown f_1(\langle w_n \rangle))\ \} \\ f \triangleq & f_1 \cup f_2 \end{aligned}$$

Enunciado Assinale a alternativa contendo uma afirmação **falsa**:

1. $f(\langle 4, 5 \rangle) = \bot$

2. $f(\langle 6 \rangle) = \langle 9 \rangle$ por aplicação direta de f_1.
3. $f(\langle 5, 4 \rangle) = \langle 8, 7 \rangle$ por aplicação recursiva de f_2.
4. $\{\langle 4 \rangle \mapsto \langle 7 \rangle, \langle 5, 4 \rangle \mapsto \langle 8, 7 \rangle, \langle 6, 5, 4, 4 \rangle \mapsto \langle 9, 8, 7, 7 \rangle\} \subset f$

Forma-λ Fubã se lembra das formas-*lambda* para a definição de funções:

$$cifraCadeia \triangleq \lambda w : \mathsf{seq} \mid (w = s \frown \langle w_n \rangle) \bullet \textbf{if } (\#w = 1) \textbf{ then } \langle cifraSimb(w_1) \rangle \textbf{ else } cifraCadeia(s) \frown \langle cifraSimb(w_n) \rangle$$

 — E é fácil implementar esta definição por um método recursivo:

```
# CifraCesar <<função de estado>>::
def cifraCadeia(w) :
  n = len(w)
  if (n == 1) :
    return cifraSimb(w[0])
  s = w[0 : n-1]
  return cifraCadeia(s) + cifraSimb(w[n-1])
# usando
cifraCadeia ("mel") # <-- "pho"
```

 — Falta modelar o deciframento de uma palavra!

8.3 Exercícios

8.3.1 Números de Fibonacci

Contexto Os números de Fibonacci são uma sequência de inteiros, cada um dos quais corresponde à soma dos dois números anteriores (Lewis & Loftus, 2011, p. 494). Os primeiros dois números da sequência são 0 e 1.

Enunciado

1. Apresente uma definição recursiva dos números de Fibonacci na forma da função fib.
2. Implemente a definição do item anterior em Python.
3. Apresente uma razão para se evitar o uso de um programa recursivo para resolver este problema.

8.3.2 Somatória

Contexto Considere a função *soma* apresentada no texto principal. Deseja-se modificá-la conforme a seguinte definição recursiva: a soma de 1 até n é a soma de 1 até $\frac{n}{2}$ mais a soma de $(\frac{n}{2} + 1)$ até n (Lewis & Loftus, 2011, p. 494). A nova função chama-se f.

Enunciado

1. Apresente a definição da função f usando uma forma-λ.
2. Calcule o valor de $f(7)$.
3. Implemente a definição de f com base na forma-λ anteriormente apresentada.

8.3.3 Decodificação de Cadeias

Desenvolver o código da função que decodifica uma cadeia de entrada. Exemplo: `"ba"`, deve ser retornado `"ed"`. O código deve ser obtido a partir de um modelo baseado na sequência A.

8.3.4 Decodificação de Cifras de César [**]

Contexto No texto principal desenvolveu-se a função $cifraCadeia$ para suportar a codificação de textos.

Enunciado

1. Desenvolva a função inversa $cifraCadeia^{-1}$.
2. Implemente o modelo do item anterior em Python.
3. Mostre que o seu programa decodifica textos cifrados pela função $cifraCadeia$.

Referências

Lewis, J., & Loftus, W. (2011). *Java Software Solutions: Foundations of Program Design*. Pearson Education. Recuperado de http://books.google.com.au/books?id=KnxiewAACAAJ