
CORREÇÃO DE PROGRAMAS - Aula 3 (Wikipédia, "Loop_variant")

Para mostrarmos a correção TOTAL de algoritmos iterativos, precisamos ser capazes de argumentar que a execução de um laço em algum momento chega ao fim. Nesse sentido, o seguinte conceito é bastante útil:

1. DEFINIÇÃO: um VARIANTE de um laço é uma função dos estados de um programa num
   --------- conjunto que possua uma ORDEM BEM-FUNDADA, função essa cujo valor assumido decresce estritamente toda vez que uma iteração do laço em questão é executada.

   (Para mais sobre os conceitos de "variante de laço" e "ordem bem-fundada", consulte a Wikipédia: referência ao final).

Informalmente, um variante de laço é uma quantidade (frequentemente associada às variáveis de um programa) que diminui progressivamente à medida em que as iterações de um laço vão se sucedendo; como essa quantidade não pode se tornar menor do que um certo limite (daí precisarmos que a ordem seja "bem-fundada"), então fica evidente que o laço não pode executar indefinidamente.

Na prática, o conjunto dos números naturais é geralmente escolhido para ser o "conjunto que possui uma ordem bem-fundada", como podemos fazer no caso do algoritmo de busca linear:

```
  ========================================================
  Algoritmo: busca_linear
  Entrada: um vetor A[1..n] e um valor "v".
  Saída: um número natural.
  --------------------------------------------------------
  1. i := 1
     // VARIANTE: n + 1 - i
  2. ENQUANTO i <= n
  3. | SE A[i] = v
  4. | | RETORNE i.
  5. | ++i
  6. RETORNE i.
  ========================================================
```

Perceba que a expressão "n +1 -i" diminui a cada iteração do laço do algoritmo, e também que ela não assume valor menor que zero; ela é, de fato, um variante do laço do algoritmo.

USANDO VARIANTES PARA VERIFICAR ALGORITMOS

Em nota de aula anterior, nós mostramos que o algoritmo de busca linear é parcialmente correto, isto é, que, nos casos em que ele termina de executar, ele retorna uma resposta correta. Para concluirmos que o algoritmo é totalmente correto, resta apenas, portanto, mostrar que ele sempre termina. Isso segue do seguinte resultado geral:

2. FATO: todo laço que possui um variante executa um número limitado de
   ---- iterações, ou seja, em cada execução, ou o laço termina de executar ou alguma das iterações do laço não chega ao fim.

Uma prova desse resultado geral deve seguir diretamente por indução transfinita. Para os casos em que o variante é uma função sobre os números naturais, deve ser fácil mostrar o resultado por indução matemática comum.

3. COROLÁRIO: se o algoritmo busca_linear possui um variante, então toda
   --------- execução desse algoritmo termina.

==========
PROVA:

Pelo fato 2 acima, se o laço do algoritmo busca_linear possui um variante, então, em cada execução do algoritmo, ou o laço termina ou então alguma das iterações do laço não chega ao fim. Esse segundo caso, porém, é impossível, pois, em uma iteração qualquer do laço, ou a condição da linha 3 é verdadeira e a iteração termina por execução da linha 4, ou a condição é falsa e a iteração termina logo após a (necessária) execução da linha 5. Assim, em toda execução do algoritmo, o laço termina, e portanto o algoritmo também, CQD.
==========

 PROVANDO VARIANTES

Neste ponto, a utilidade do conceito de variante de laço deve estar evidente. Resta, porém, saber como argumentar que uma função é de fato um variante de laço. A estratégia abaixo segue diretamente da definição do conceito:

  * PASSO 1. VALOR INICIAL: mostrar que, no início do laço, a função em questão
    ---------------------- assume um valor em um certo conjunto "C" (que possua uma relação de ordem bem-fundada).

  * PASSO 2. DIMINUIÇÃO: mostrar que, após uma iteração qualquer que chegue ao
    ------------------ fim e depois da qual a condição do laço é novamente avaliada, a função assume um valor no conjunto "C" que é estritamente menor do que o valor assumido pela função no início da iteração.

Exemplo:

4. LEMA: "n +1 -i" é um variante do laço do algoritmo busca_linear.

==========
PROVA:

1. VALOR INICIAL: no início do laço, vale i = 1 e n >= 0, ou seja, n +1 -i >= 0.
   ------------- Logo, "n +1 -i" é um número natural.

2. DIMINUIÇÃO: considere uma iteração qualquer do laço, após a qual a condição
   ---------- do laço é novamente avaliada. Seja "x" o valor de "i" no início da iteração em questão. Logo, no início da iteração em questão, a expressão "n +1 -i" assume o valor "n +1 -x". Agora, como a linha 4 não é executada nessa iteração, então "i" é incrementada e assume o valor "x+1" ao final da iteração, quando então a expressão "n +1 -i" assume o valor "n - x". Logo, a expressão "n +1 -i" assume, ao final da iteração, um valor inteiro estritamente menor que aquele do início da iteração. Finalmente, como i <= n+1 é um invariante do laço, então, ao final da iteração em questão, vale n +1 -i >= 0, e portanto o valor assumido pela expressão "n +1 -i" é, na ocasião, um número natural, CQD.

==========

5. EXERCÍCIOS:

   a) Prove a correção total da versão iterativa do algoritmo de busca binária.

   b) Prove a correção total do algoritmo de MDC de Euclides
      (RESPOSTA: CAnA 2014-2, AP1, Q1).

c) Prove a correção total do algoritmo abaixo

```
=========================================================
Algoritmo: máx
Entrada: um vetor A[1..n].  // n >= 1
Saída: um elemento do vetor.
---------------------------------------------------------
1. m := A[1] ; i := 2
   // VARIANTE: n -i +1
   // INVARIANTES: * 2 <= i <= n+1.
   //              * ∀j ∈ {1..i-1}, m >= A[j].
   //              * ∃j ∈ {1..i-1} tal que A[j] = m.
2. ENQUANTO i <= n
3. | SE A[i] > m
4. | | m := A[i]
5. | ++i
6. RETORNE m.
=========================================================
```

com relação à seguinte especificação:

* Toda chamada máx(A) termina e retorna um valor "k" tal que

  a) ∀j ∈ {1..n}, A[j] <= k.
  b) ∃j ∈ {1..n} tal que A[j] = k.

6. REFERÊNCIAS:

a) Sobre terminação de algoritmos e variantes de laço:

* http://en.wikipedia.org/wiki/Termination_analysis
* http://en.wikipedia.org/wiki/Loop_variant