Invariantes de Laço

Profa. Sheila Morais de Almeida

DAINF-UTFPR-PG

junho - 2018

Este material é preparado usando como referências os textos dos seguintes livros.

Thomas H. CORMEN, Charles E. LEISERSON, Ronald L. RIVEST, Clifford STEIN. **Introduction to Algorithms**, 2nd ed., 2001.

Judith L. GERSTING, **Mathematical Structures For Computer Science: A Modern Approach to Discrete Mathematics**, 6th ed., 2007.

Kenneth ROSEN. **Discrete Mathematics and Its Applications**. McGraw-Hill Education, 6th edition (July 26, 2006).

Udi Manber. **Introduction to Algorithms: a creative approach.**, 1st ed., 1989.

Invariantes de laço

A invariante de laço pode ser definida como uma relação entre as variáveis de um algoritmo que é verdadeira em um determinado ponto p (ou linha p) do laço.

Como é algo invariante, essa relação precisa ser verdadeira:

- antes de se executar os comandos da linha p pela primeira vez;
- durante a execução do laço, em todas as iterações, antes da execução da linha p;
- no fim da execução do laço, após a última execução da linha p.

Invariantes de laço

Considere o seguinte algoritmo que computa a soma dos elementos de um vetor de inteiros:

```
1   int soma(int A)
2       s ← 0
3       i ← 1
4       while i ≤ A.lenght do
5           s ← s + A[i]
6           i ← i + 1
7       return s
```

Invariantes de laço

Considere o seguinte algoritmo que computa a soma dos elementos de um vetor de inteiros:

```
1   int soma(int A)
2       s ← 0
3       i ← 1
4       while i ≤ A.lenght do
5           s ← s + A[i]
6           i ← i + 1
7       return s
```

Invariante: No início de cada iteração do laço, a variável s contém a soma dos elementos do vetor A da posição 1 à posição $i - 1$.

Invariantes de laço

Considere o seguinte algoritmo que computa a potência x^n, para inteiros x e n:

```
1   int potencia(int x, int n)
2       p ← 1
3       i ← 0
4       while i < n do
5           p ← p × x
6           i ← i + 1
7       return p
```

Invariantes de laço

Considere o seguinte algoritmo que computa a potência x^n, para inteiros x e n:

```
1   int potencia(int x, int n)
2       p ← 1
3       i ← 0
4       while i < n do
5           p ← p × x
6           i ← i + 1
7       return p
```

Invariante: No início de cada iteração do laço, a variável p é igual a x^i.

Invariantes de laço

Invariantes são utilizadas para provar a corretude de algoritmos iterativos.

Os passos para tais provas são:

1. Identificar os laços e analisar um laço por vez, começando do mais interno.

2. Determinar uma invariante para cada laço: uma relação entre as variáveis que permanece verdadeira em todas as iterações e que é capaz de descrever qual o progresso feito pelo laço a cada iteração.

    - (Achar a invariante é quase sempre a parte mais difícil.)

Invariantes de laço

3. Provar as invariantes: provar que as invariantes dos laços são verdadeiras.

4. Usar as invariantes dos laços para provar que o algoritmo termina.

5. Provar corretude: usar as invariantes dos laços para provar que o algoritmo obtém o resultado correto.

Invariantes de laço

```
1   int fibonacci(int n)
2       Se n = 0 então
3           retorna 0
4       a ← 0; b ← 1; i ← 2
5       while i ≤ n do
6           c ← a + b;
7           a ← b;
8           b ← c; i ← i + 1
9       return b
```

Invariantes de laço

```
1   int fibonacci(int n)
2       Se n = 0 então
3           retorna 0
4       a ← 0; b ← 1; i ← 2
5       while i ≤ n do
6           c ← a + b;
7           a ← b;
8           b ← c; i ← i + 1
9       return b
```

Invariante: Para todo natural $j \geq 0$, no início da iteração $j+1$ (linha 5), $i = j+2$, $a = fibonacci(j)$ e $b = fibonacci(j+1)$.

Invariantes de laço

Fatos sobre o algoritmo fibonacci(n):

- no início da primeira iteração do laço (linha 5), $i = 2$.
- no início da iteração $j + 1$ (linha 5), $i = j + 1$.
- no início da primeira iteração do laço (linha 5), $a = 0$.
- no início da iteração $j + 1$ (linha 5), $a = b$.
- no início da primeira iteração do laço (linha 5), $b = 1$.

Invariantes de laço

Fatos sobre o algoritmo fibonacci(n):

- no início da primeira iteração do laço (linha 5), $i = 2$. Ok!
- no início da iteração $j + 1$ (linha 5), $i = j + 1$. Ok!
- no início da primeira iteração do laço (linha 5), $a = 0$.Ok!
- no início da iteração $j + 1$ (linha 5), $a = b$. Não!
- no início da primeira iteração do laço (linha 5), $b = 1$. Ok!

Invariantes de laço

Fatos sobre o algoritmo fibonacci(n):

- no início da iteração $j + 1$ (linha 5), $b = c$.
- No início de qualquer iteração (linha 5) $c = a + b$.
- Em qualquer iteração, na linha 6, $c = a + b$.

Invariantes de laço

Fatos sobre o algoritmo fibonacci(n):

- no início da iteração $j + 1$ (linha 5), $b = c$. não!
- No início de qualquer iteração (linha 5) $c = a + b$. Não!
- Em qualquer iteração, na linha 6, $c = a + b$. Ok!

Invariantes de laço

Teorema da Invariante: Para todo natural $j \geq 0$, no início da iteração $j+1$ (linha 5), $i = j+2$, $a = \mathit{fibonacci}(j)$ e $b = \mathit{fibonacci}(j+1)$.

Demonstração: A demonstração é por indução em j.

Base: para $j = 0$
$i = 0+2 = 2$, $a = 0 = \mathit{fibonacci}(0)$ e $b = 1 = \mathit{fibonacci}(1)$.

Hipótese: Para um inteiro $0 \leq k < n$, no incício da iteração $k+1$ (linha 5), tem-se:

$i = k+2$, $a = \mathit{fibonacci}(k)$ e $b = \mathit{fibonacci}(k+1)$.

Invariantes de laço

Passo: Queremos mostrar que no início da iteração $k+2$, linha 5:

$i = k+3$, $a = fibonacci(k+1)$ e $b = fibonacci(k+2)$.

No início da $k+1$-ésima iteração, $i = k+2$ (por hipótese de indução).

A variável i só é alterada na linha 8, quando é incrementada pelo comando $i \leftarrow i+1$.

Então no início da iteração $k+2$, $i = k+2+1 = k+3$.

Invariantes de laço

Passo: Queremos mostrar que no início da iteração $k+2$, linha 5:

$i = k+3$, $a = fibonacci(k+1)$ e $b = fibonacci(k+2)$.

Na $k+1$-ésima iteração, linha 7, $a \leftarrow b$.

Por hipótese de indução, no início da iteração, $b = fibonacci(k+1)$

e b não se alterou nas linhas 5 e 6.

Então, na linha 7, tem-se $a = b = fibonacci(k+1)$.

Como a não é alterada na linha 8, tem-se $a = fibonacci(k+1)$ no início da iteração $k+1$.

Invariantes de laço

Passo: Queremos mostrar que no início da iteração $k+2$, linha 5:

$i = k+3$, $a = fibonacci(k+1)$ e $b = fibonacci(k+2)$.

Na $k+1$-ésima iteração, linha 6, $c \leftarrow a+b$.

Como os valores de a e b não se alteram nas linhas 5 e 6,
por hipótese de indução na linha 6, tem-se:

$$c = a + b = fibonacci(k) + fibonacci(k+1) = fibonacci(k+2).$$

Na linha 8, $b \leftarrow c$. Logo, na k-ésima iteração da linha 8,
$b = fibonacci(k+2)$.

O valor de b não se altera até o início da iteração $k+1$, quando
$b = fibonacci(k+2)$. $\square$

Invariantes de laço

Prova de Corretude

Teorema: O algoritmo retorna o valor de $fibonacci(n)$.

Demonstração: A afirmação é claramente verdadeira se $n = 0$ pelas linhas 2 e 3.

Se $n > 0$, então entramos no laço.

Na última iteração do laço, após a linha 8, i é igual a $n + 1$ e o laço termina.

Pelo Teorema da Invariante, no início da iteração $j + 1$, $i = j + 2$. Então, $j + 2 = n + 1$.

Então, $j = n - 1$. Portanto, ocorreram $n - 1$ iterações.

Prova de Corretude

Teorema: O algoritmo retorna o valor de $fibonacci(n)$.

Pelo Teorema da Invariante, no início da iteração n, tem-se $j + 1 = n$ e $b = fibonacci(j + 1) = fibonacci(n)$.

Como a condição para a execução do laço é falsa quando $i = n + 1$, então o laço termina e o algoritmo retorna $b = fibonacci(n)$. $\square$

Multiplicação iterativa

```
1  Algoritmo multiplica(y, z)
2      x ← 0
3      enquanto z > 0 faça:
4          se z é ímpar então,
5              x ← x + y
6          y ← 2y
7          z ← ⌊z/2⌋
8      retorna x
```

Multiplicação iterativa

1 **Algoritmo multiplica**(y, z)
2 $x \leftarrow 0$
3 **enquanto** $z > 0$ **faça:**
4 **se** z **é ímpar então,**
5 $x \leftarrow x + y$
6 $y \leftarrow 2y$
7 $z \leftarrow \lfloor \frac{z}{2} \rfloor$
8 **retorna** x

Teorema: Se y, $z \in \mathbb{N}$, o Algoritmo multiplica(y,z) retorna $y \times z$.

Multiplicação iterativa

Resultado preliminar:

Teorema: Para todo natural n, $2\lfloor \frac{n}{2} \rfloor + (n mod 2) = n$.

Demonstração: Há dois casos:

Caso 1: n é par.

Então $\lfloor \frac{n}{2} \rfloor = \frac{n}{2}$ e $(n mod 2) = 0$ e o resultado segue.

Multiplicação iterativa

Resultado preliminar:

Teorema: Para todo natural n, $2\lfloor \frac{n}{2} \rfloor + (n mod 2) = n$.

Caso 2: n é ímpar.

Então $\lfloor \frac{n}{2} \rfloor = \frac{n-1}{2}$ e $(n mod 2) = 1$ e o resultado segue. $\square$

Multiplicação iterativa

Para todo natural $j \leq 0$, sejam y_j, z_j e x_j os valores das variáveis x, y e z no início de qualquer iteração j (na linha 3).

Invariante do laço:

Teorema: Para todo inteiro $j \geq 0$, $y_j z_j + x_j = y_0 z_0$.

Demonstração: A prova é por indução em j.

Base: $j = 0$: Nesse caso, $x_0 = 0$, então $y_0 z_0 + x_0 = y_0 z_0$

Multiplicação iterativa

Hipótese de indução: Para um inteiro não-negativo k, $y_k z_k + x_k = y_0 z_0$.

Passo: Observe que $x_{k+1} = x_k + y_k(z_k mod 2)$, pelas linhas 4 e 5.

Então, $y_{k+1} z_{k+1} + x_{k+1} = 2y_k \lfloor z_k 2 \rfloor + x_k + y_k(z_k mod 2) =$

$= y_k(2\lfloor z_k 2 \rfloor + (z_k mod 2)) + x_k =$ (pelo resultado preliminar)

$= y_k z_k + x_k =$ (pela hipotese de indução)

$= y_0 z_0$ $\square$

Multiplicação iterativa

Prova de corretude

Teorema: O Algoritmo multiplica(y,z) retorna $y \times z$.

prova: O algoritmo termina, pois em cada iteração z é dividido pela metade e arrendondado para baixo se for ímpar. Logo, para alguma iteração t, $z_t = 0$.

Pela linha 8, o algoritmo retorna x_t.

Pela invariante do laço, $y_t z_t + x_t = y_0 z_0$.

Como $z_t = 0$, tem-se $x_t = y_0 z_0 = yz$ $\square$