Universidade Federal do Ceará – Campus Quixadá

QXD0041 – Projeto e Análise de Algoritmos – 1º Semestre de 2024

Lista de Exercícios

Projeto, Correção e Análise de Algoritmos Iterativos

Invariantes de laço e demonstração de correção

Questão 1. (CLRS) Exercício 2.1-3. Considere o seguinte problema de busca.

Input: Uma sequência de n números $A = \langle a_1, a_2, \ldots, a_n \rangle$ e um valor v.

Output: Um índice i tal que $v = A[i]$, ou o valor especial `NIL` se v não aparece em A.

Escreva pseudocódigo para a **busca linear**, que varre a sequência A procurando por v. Usando uma invariante de laço, prove que seu algoritmo está correto. Certifique-se de que sua invariante de laço atende às três propriedades necessárias. Além disso, argumente que a complexidade de pior caso da busca linear é $\Theta(n)$.

Questão 2. (CLRS) Exercício 2.3-5. Voltando ao problema de busca da questão anterior, observe que se a sequência A dada como entrada estiver ordenada em ordem crescente, então podemos verificar o ponto médio da sequência e eliminar metade da sequência de considerações posteriores. O algoritmo de **busca binária** repete este procedimento, reduzindo pela metade o tamanho restante da sequência A a cada iteração. Escreva o pseudocódigo de um algoritmo iterativo para a busca binária. Argumente que o tempo de execução de pior caso da busca binária é $\Theta(\lg n)$.

Questão 3. (CLRS) Problema 2-2 **(Corretude do BubbleSort)**. Bubblesort é um algoritmo de ordenação popular, mas ineficiente. Funciona repetidamente trocando pares de elementos adjacentes que estão fora de ordem.

```
BUBBLESORT(A)
1  for i = 1 to A.length - 1
2      for j = A.length downto i + 1
3          if A[j] < A[j - 1]
4              exchange A[j] with A[j - 1]
```

(a) Seja A' a saída do algoritmo BUBBLESORT. A fim de provar que o BUBBLESORT é correto, precisamos provar que ele termina e que

$$A'[1] \leq A'[2] \leq \cdots \leq A'[n] \tag{1}$$

onde $n = A.length$. A fim de mostrar que o BUBBLESORT de fato ordena o arranjo dado como entrada, o que mais precisamos mostrar?

Os próximos dois itens provam a Equação (1).

(b) Declare de forma precisa uma invariante de laço para o laço **for** das linhas 2–4 e prove que essa invariante de laço é verdadeira. Você deve usar indução matemática a fim de provar os três passos da prova de invariante de laço: Inicialização, Manutenção e Término.

(c) Usando a Condição de Término da invariante de laço provada no item (b) anterior, declare uma invariante de laço para o laço das linhas 1–4 que permita a você provar a desigualdade 1. Sua prova deveria usar a estrutura de prova de invariante de laço: Inicialização, Manutenção e Término.

(d) Qual é o tempo de execução do pior caso do BUBBLESORT? Como ele se compara ao tempo de execução de pior caso do INSERTIONSORT?

Questão 4. (Manber) (Exercício 2.40) O Algoritmo CONVERT_TO_BINARY, descrito abaixo, converte um inteiro n para binário. Modifique o algoritmo CONVERT_TO_BINARY de tal forma que ele converta um número dado em base 6 para um número binário. A entrada é um vetor de dígitos na base 6 e a saída é um vetor de bits. Mostre a correção de seu algoritmo utilizando uma invariante de laço.

```
Algorithm Convert_to_Binary (n) ;
Input: n (a positive integer).
Output: b (an array of bits corresponding to the binary representation of n).

begin
    t := n ; { we use a new variable t to preserve n }
    k := 0 ;
    while t > 0 do
        k := k + 1 ;
        b[k] := t mod 2 ;
        t := t div 2 ;
end
```

Figura 1: Algoritmo que converte um inteiro positivo n para binário.

Questão 5. Considere o Algoritmo de Euclides a seguir, que determina o máximo divisor comum entre dois números naturais:

Algoritmo 1 Algoritmo de Euclides

```
Procedimento EUCLIDES(a, b)                      ▷ Obtém o MDC de a e b
    r ← a mod b
    enquanto r ≠ 0 faça                          ▷ Já sabemos a resposta se r é 0
        a ← b
        b ← r
        r ← a mod b
    devolve b                                    ▷ O MDC é b
```

Você deve mostrar formalmente que o algoritmo está correto utilizando uma invariante de laço.

(a) Escreva uma invariante de laço adequada para o algoritmo.
(b) Demonstre a invariante de laço.
(c) Demonstre que o algoritmo está correto utilizando a afirmação acima.

Questão 6. Seja $f(x)$ uma função real contínua e suponha que ela tem uma ou mais raízes entre a, b (uma raiz é um número $r \in (a,b)$ com $f(r) = 0$). Dado $\varepsilon > 0$, uma ε-aproximação de uma raiz, é um número $x \in (a,b)$ tal que $x \in (r-\varepsilon, r+\varepsilon)$ em que r é uma raiz. O método de aproximação da raiz baseado em busca binária é descrito no algoritmo a seguir:

```
Bisect(a, b, ε)
    enquanto b − a > ε:
        se f((a+b)/2) ≤ 0:
            a ← (a+b)/2
        senão:
            b ← (a+b)/2
    devolva (a+b)/2
```

(a) Analise o tempo de execução desse algoritmo em termos $b - a$ e ε.

(b) Dê condições (suficientes) sobre o entrada para que o algoritmo termine com uma resposta correta. Depois demonstre que o algoritmo está correto (quando dada uma entrada válida).