Exercícios - Análise de Algoritmos Recursivos

Prof. André Vignatti

Exercício 1. Mostre passo-a-passo como o `MergeSort` executa no vetor $(3, 41, 52, 26, 38, 57, 9, 49)$.

Exercício 2. Considere o algoritmo Intercala apresentado em aula.

Afirmação (Invariante do Intercala)**.** No começo de cada iteração do laço das linhas 7–12, vale que:

1. $A[p \dots k-1]$ está ordenado,
2. $A[p \dots k-1]$ contém todos os elementos de $B[p \dots i-1]$ e de $B[j+1 \dots r]$,
3. $B[i] \geq A[k-1]$ e $B[j] \geq A[k-1]$.

(a) Prove que a afirmação acima é de fato um invariante de Intercala.

(b) (fácil) Mostre usando o invariante acima que Intercala é correto.

Exercício 3. Supondo que o algoritmo Intercala está correto (resolvido no Exercício 2), prove que o Mergesort está correto.

Exercício 4. Considere o seguinte problema de busca:

Entrada: Uma seqüência de n números $A = (a_1, a_2, \dots, a_n)$ e um valor v.
Saída: Um índice i tal que $v = A[i]$ ou o valor especial `NIL` se v não aparece em A.

Note que, se o vetor A está ordenado, podemos comparar o elemento $A[n/2]$ com v, eliminando metade do vetor em comparações futuras. Tal procedimento é chamado de `Busca Binária`. Escreva um pseudo-código (recursivo ou iterativo) para a `Busca Binária` e argumente que o tempo execução de pior-caso é $\Theta(\log n)$.

Exercício 5. Observe que o laço interno do `InsertionSort` usa uma busca linear para varrer (de trás para frente) o subvetor ordenado $A[1..j-1]$. Podemos usar a `Busca Binária` (do Exercício 4) para tentar melhorar o pior-caso do `InsertionSort` para $\Theta(n \log n)$?