Complexidade de Algoritmos – Prova 1

Turma 2016 e 2017 Data: 15/10/2018 Prof. Manoel Ribeiro

1. Um algoritmo **a** tem complexidade $2n^2$ e o algoritmo **b** complexidade 4^n. Num certo computador, num tempo t, o algoritmo **a** resolve um problema de tamanho x e o algoritmo **b** um problema de tamanho y. Imagine agora que você tem disponível um computador 32 vezes mais rápido. Que tamanho de problema resolverão os algoritmos **a** e **b**, no mesmo tempo t? Analise a resposta. (1,5)

Algoritmo a

$C = 2n^2$ $T = t$ $n = x$ $Vc = v$

Algoritmo b

$C = 4^n$ $T = t$ $n = y$ $Vc = v$

Agora $Vc = 32v$

Algoritmo a

$t = 2x^2/v = 2n^2/32v$

$n^2 = 32\ x^2$

$n = 5{,}65\ x$

Algoritmo b

$t = 4^y/v = 4^n\ /32v$

$4^n = 324^y$

$(2^2)^n = 2^5\ (2^2)^y$

$2^{2n} = 2^5 2^{2y}$

$2^{2n} = 2^{5+2y}$

$2n = 5 + 2y$

$n = 2{,}5 + y$

Com o aumento da velocidade do computador o algoritmo a consegue resolver um problema mais do que 5 vezes maior, por exemplo para um problema de tamanho 10 conseguirá resolver um problema maior que 50, 56 vezes maior. Enquanto o algoritmo b, em um computador com o mesmo aumento de velocidade só conseguirá soma mais 2,5. Para um problema de tamanho 10 só resolvera um problema de tamanho 12,5. O que demonstra que o algoritmo a é mais eficiente que o algoritmo b.

2. Sejam dois algoritmos A e B com complexidade $500n^2$ e n^5. Analise o tempo de resposta desses dois algoritmos. (1,5)

Valor de n	$500n^2$	n^5
1	500	1
5	12500	3125
7	24500	16807
8	32000	32768
9	40500	59049
10	50000	100000

Para n< =7 o algoritmo b é mais eficiente
Para n > =8 o algoritmo a é mais eficiente

3. Escreva o pseudocódigo de um algoritmo que retorne o valor máximo contido em um arranjo A de n posições (1,5). Qual invariante de laço esse algoritmo mantém? Usando um invariante de laço, prove que seu algoritmo é correto. Certifique-se de que seu invariante de laço satisfaz as três propriedades necessárias (1,0). Para esse algoritmo forneça os tempos de execução do melhor caso e do pior em notação Θ (1,0).

```
Algoritmo Máximo(A,n)
i=1
x = A[i]
for i = 2 to n
      if ( V[i] > x
            x= V[i]
return x
```

A **invariante de laço** para esse pseudocódigo é a seguinte: Sempre a variável x contém o maior valor do subarray A[1 ...i-1]

Inicialização: Antes de entrar no laço x= A[1], que é maior valor já que i=1 e A[1 ...i-1] só tem o valor A[1].

Manutenção: A cada volta do Loop, claramente x tem o maior valor de A[1.. i-1]

Témino: Quando sair do laço i=n+1 e a variável x tem o maior valor de do vetor A[1.. n].

Para esse algoritmo forneça os tempos de execução do melhor caso e do pior em notação Θ

Algoritmo Máximo(A,n)	custo	vezes
0 i=1	c_o	1
1 x = A[i]	c_1	1
2 for i = 2 to n	c_2	n
3 if (V[i] > x	c_3	n-1
4 x= V[i]	c_4	n-1
5 return x	c_5	1

$T(n) = c_o + c_1 + c_2 n + c_3(n-1) + c_4(n-1) + c_5$

No melhor caso, o valor máximo está em A[1], portanto c_4 é igual a zero, e $T(n) = (c_2 + c_3)n + c_0 + c1 - c_3 + c_5$, o que significa que a complexidade é Θ(n).

No pior caso $T(n) = (c_2 + c_3 + c_4)n + c_0 + c1 - c_3 - c_4 + c_5$, portanto a complexidade também é Θ(n).

4. Conhecendo-se os valores da variação diária de temperatura num determinado lugar ao longo de um certo tempo (10, 50 ou 100 anos por exemplo), queremos encontrar uma sequência de dias em que a variação acumulada tenha sido máxima. Como exemplo a variação de temperatura ao longo de oito dias poderia ter sido, em décimos de grau

   20 -30 15 -10 30 -20 -30 30

   Esse problema pode ser resolvido pelo cálculo da altura de um vetor A[1 . . . n], que é a soma de um segmento de soma máxima, por exemplo a altura do vetor do exemplo acima é 15 -10 +30 = 35.

   Um **segmento** de um vetor A[1 . . . n] é qualquer subvetor da forma A[i . . . k], com 1 <= i <= k <= n. A condição i < k garante que segmentos não são vazios.1 A **soma** de um segmento A[i ... k] é o número A[i] + A[i + 1] + - - -+ A[k]. Escreva um algoritmo que calcule a altura de um vetor A[1 . . . n] de números inteiros (1,5). Qual invariante de laço esse algoritmo mantém? Usando um invariante de laço, prove que seu algoritmo é correto. Certifique-se de que seu invariante de laço satisfaz as três propriedades necessárias (1,0). Para esse algoritmo forneça os tempos de execução do melhor caso e do pior em notação Θ (1,0).

O algoritmo óbvio para o problema do segmento de soma máxima examina, sistematicamente, todos os segmentos de A[1 ...n] e escolhe o que tiver maior soma.

```
Algoritmo AlturaVetor (A, n)
1 x = A[1]
2 para i = 1 até n
3       para k = i até n
4             s = 0
5             para j = i até k
6                   s  = s + A[j]
7             se s > x então x =   s
8 retorna  x
```

Mostrar o do Max que está igual a esse.

O algoritmo está correto. No início de cada execução da linha 7, s é a soma do segmento A[i ... k]. Como i varia de 1 até n e k varia de i até n, o valor de x na linha 8 é a altura do vetor A[1 ...n].

Versão do Antônio Carlos

```
Algoritmo altura_Segmento(array A[n])
1Máximo = 0
2 For i = 1 to (n)
3       For j = i to (n)
4             S = 0
5             For m = i to (j)
6                   S = S + A[m]
7       If(S>Maximo)
8             Maximo = S
9Retorna Maximo
```

Mostrar versão do Fabricio e o Kristhyan que são semelhantes.

O do Mário está semelhante ao dos Fabricio e Kristhyan, mas está errado.