Aula 2

Divisão e conquista

Exemplo 1: Número de inversões de uma permutação
(problema 2-4 do CLRS; veja também sec 5.4 do KT)

Exemplo 2: Par de pontos mais próximos
(sec 33.4 do CLRS)

Essas transparências foram adaptadas das transparências do Prof. Paulo Feofiloff e do Prof. José Coelho de Pina.

Número de inversões

Problema: Dada uma permutação $p[1\,..\,n]$, determinar o número de inversões em p.

Inversão: par (i, j) de índices de p tal que $i < j$ e $p[i] > p[j]$.

Número de inversões

Problema: Dada uma permutação $p[1\,..\,n]$, determinar o número de inversões em p.

Inversão: par (i, j) de índices de p tal que $i < j$ e $p[i] > p[j]$.

Queremos um algoritmo $\mathrm{O}(n \lg n)$ para o problema.

O número de inversões pode ser $\Theta(n^2)$.

Portanto, não podemos contar de uma em uma as inversões, como faz o algoritmo visto na aula passada.

Número de inversões

Problema: Dada uma permutação $p[1 \,..\, n]$, determinar o número de inversões em p.

Inversão: par (i, j) de índices de p tal que $i < j$ e $p[i] > p[j]$.

Queremos um algoritmo $\mathrm{O}(n \lg n)$ para o problema.

O número de inversões pode ser $\Theta(n^2)$.

Portanto, não podemos contar de uma em uma as inversões, como faz o algoritmo visto na aula passada.

Idéia: Vamos ordenar e contar ao mesmo tempo!

A ordenação ajuda a contar várias inversões de uma só vez.

Que algoritmo de ordenação usaremos?

Duas opções: o MERGESORT e o HEAPSORT. Qual deles parece mais adequado?

Número de inversões

Problema: Dada uma permutação $p[1 \mathinner{.\,.} n]$, determinar o número de inversões em p.

Inversão: par (i, j) de índices de p tal que $i < j$ e $p[i] > p[j]$.

Queremos um algoritmo $\mathrm{O}(n \lg n)$ para o problema.

O número de inversões pode ser $\Theta(n^2)$.

Portanto, não podemos contar de uma em uma as inversões, como faz o algoritmo visto na aula passada.

Idéia: Vamos ordenar e contar ao mesmo tempo!

A ordenação ajuda a contar várias inversões de uma só vez.

Que algoritmo de ordenação usaremos?

Duas opções: o MERGESORT e o HEAPSORT. Qual deles parece mais adequado?

Resposta: o MERGESORT.

Merge-Sort

Rearranja $A[p\,..\,r]$, com $p \leq r$, em ordem crescente.

MERGESORT (A, p, r)
se $p < r$
  então $q \leftarrow \lfloor (p+r)/2 \rfloor$
    MERGESORT (A, p, q)
    MERGESORT $(A, q+1, r)$
    INTERCALA (A, p, q, r)

Método: Divisão e conquista.

Intercalação

Problema: Dados $A[p\,..\,q]$ e $A[q{+}1\,..\,r]$ crescentes, rearranjar $A[p\,..\,r]$ de modo que ele fique em ordem crescente.

Para que valores de q o problema faz sentido?

Entra:

$$
\begin{array}{c c|c|c|c|c|c|c|c|c|c|}
 & & p & & & & q & & & & r \\
\cline{3-11}
A & & 22 & 33 & 55 & 77 & 99 & 11 & 44 & 66 & 88 \\
\cline{3-11}
\end{array}
$$

Intercalação

Problema: Dados $A[p \,.\,.\, q]$ e $A[q+1 \,.\,.\, r]$ crescentes, rearranjar $A[p \,.\,.\, r]$ de modo que ele fique em ordem crescente.

Para que valores de q o problema faz sentido?

Entra:

	p				q				r
A	22	33	55	77	99	11	44	66	88

Sai:

	p				q				r
A	11	22	33	44	55	66	77	88	99

Intercalação

INTERCALA (A, p, q, r)

$\triangleright$ $B[p\,.\,.\,r]$ é um vetor auxiliar
para $i \leftarrow p$ **até** q **faça**
    $B[i] \leftarrow A[i]$
para $j \leftarrow q+1$ **até** r **faça**
    $B[r+q+1-j] \leftarrow A[j]$
$i \leftarrow p$
$j \leftarrow r$
para $k \leftarrow p$ **até** r **faça**
    se $B[i] \leq B[j]$
        então $A[k] \leftarrow B[i]$
                $i \leftarrow i+1$
        senão $A[k] \leftarrow B[j]$
                $j \leftarrow j-1$

Adaptação do Merge-Sort

Conta o número de inversões de $A[p\mathrel{.\,.}r]$, com $p \le r$, e rearranja $A[p\mathrel{.\,.}r]$ em ordem crescente.

CONTA-E-ORDENA (A, p, r)
1 **se** $p \ge r$
2 **então devolva** 0
3 **senão** $q \leftarrow \lfloor (p+r)/2 \rfloor$
4 $c \leftarrow$ CONTA-E-ORDENA $(A, p, q) +$
5 CONTA-E-ORDENA $(A, q+1, r) +$
6 CONTA-E-INTERCALA (A, p, q, r)
7 **devolva** c

Método: Divisão e conquista.

Contagem na intercalação

CONTA-E-INTERCALA (A, p, q, r)

para $i \leftarrow p$ **até** q **faça**
    $B[i] \leftarrow A[i]$
para $j \leftarrow q+1$ **até** r **faça**
    $B[r+q+1-j] \leftarrow A[j]$
$i \leftarrow p$
$j \leftarrow r$
$c \leftarrow 0$ ▷ inicializa o contador
para $k \leftarrow p$ **até** r **faça**
    se $B[i] \leq B[j]$
        então $A[k] \leftarrow B[i]$
            $i \leftarrow i+1$
        senão $A[k] \leftarrow B[j]$
            $j \leftarrow j-1$
            $c \leftarrow c + (q - i + 1)$ ▷ conta inversões
devolva c

Simulação

	p				q				r
A	22	33	55	77	99	11	44	66	88
B									

$c = 0$

Simulação

	k								
A									

	i								j
B	22	33	55	77	99	88	66	44	11

$c = 0$

Simulação

		k							
A	11								

	i							j	
B	22	33	55	77	99	88	66	44	11

$$c = 0 + 5 = 5$$

Simulação

			k							
A	11	22								

		i						j	
B	22	33	55	77	99	88	66	44	11

$c = 5$

Simulação

	k									
A	11	22	33							

	i							j	
B	22	33	55	77	99	88	66	44	11

$c = 5$

Simulação

		k								
A	11	22	33	44						

			i				j		
B	22	33	55	77	99	88	66	44	11

$c = 5 + 3 = 8$

Simulação

						k			
A	11	22	33	44	55				

				i			j		
B	22	33	55	77	99	88	66	44	11

$c = 8$

Simulação

							k		
A	11	22	33	44	55	66			

				i		j			
B	22	33	55	77	99	88	66	44	11

$$c = 8 + 2 = 10$$

Simulação

								k	
A	11	22	33	44	55	66	77		

					i	j			
B	22	33	55	77	99	88	66	44	11

$c = 10$

Simulação

k (above the last cell of A)

A

11	22	33	44	55	66	77	88	

$i = j$

B

22	33	55	77	99	88	66	44	11

$$c = 10 + 1 = 11$$

Simulação

A	11	22	33	44	55	66	77	88	99

					j	i			
B	22	33	55	77	99	88	66	44	11

$c = 11$

Contagem na intercalação

CONTA-E-INTERCALA (A, p, q, r)

para $i \leftarrow p$ **até** q **faça**
    $B[i] \leftarrow A[i]$
para $j \leftarrow q + 1$ **até** r **faça**
    $B[r + q + 1 - j] \leftarrow A[j]$
$i \leftarrow p$
$j \leftarrow r$
$c \leftarrow 0$ $\triangleright$ inicializa o contador
para $k \leftarrow p$ **até** r **faça**
    se $B[i] \leq B[j]$
        então $A[k] \leftarrow B[i]$
            $i \leftarrow i + 1$
        senão $A[k] \leftarrow B[j]$
            $j \leftarrow j - 1$
            $c \leftarrow c + (q - i + 1)$ $\triangleright$ conta inversões
devolva c

Consumo de tempo

Quanto tempo consome em função de $n := r - p + 1$?

linha	consumo de todas as execuções da linha
1	$\mathrm{O}(n)$
2	$\mathrm{O}(n)$
3	$\mathrm{O}(n)$
4	$\mathrm{O}(n)$
5–7	$\mathrm{O}(1)$
8	$\mathrm{O}(n)$
9	$\mathrm{O}(n)$
10–14	$\mathrm{O}(n)$
15	$\mathrm{O}(1)$
total	$\mathrm{O}(7n + 2)$ = $\mathrm{O}(n)$

Conclusão

O algoritmo CONTA-E-INTERCALA consome
$\mathrm{O}(n)$ unidades de tempo.

Também escreve-se

O algoritmo CONTA-E-INTERCALA consome
tempo $\mathrm{O}(n)$.

Análise do Conta-E-Ordena

Seja $T(n)$ o tempo consumido pelo CONTA-E-ORDENA.

Análise do Conta-E-Ordena

Seja $T(n)$ o tempo consumido pelo CONTA-E-ORDENA.

Vale a seguinte recorrência para $T(n)$:

$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \mathrm{O}(n)$$

Análise do Conta-E-Ordena

Seja $T(n)$ o tempo consumido pelo CONTA-E-ORDENA.

Vale a seguinte recorrência para $T(n)$:

$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \mathrm{O}(n)$$

Solução: $T(n) = \mathrm{O}(n \lg n)$.

Prova?

Análise do Conta-E-Ordena

Seja $T(n)$ o tempo consumido pelo CONTA-E-ORDENA.

Vale a seguinte recorrência para $T(n)$:

$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \mathrm{O}(n)$$

Solução: $T(n) = \mathrm{O}(n \lg n)$.

Prova?

Considera-se a recorrência simplificada

$$T(n) = 2\,T(n/2) + n$$

definida apenas para n potência de 2.

Análise do Conta-E-Ordena

Seja $T(n)$ o tempo consumido pelo CONTA-E-ORDENA.

Vale a seguinte recorrência para $T(n)$:

$$T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \mathrm{O}(n)$$

Solução: $T(n) = \mathrm{O}(n \lg n)$.

Prova?

Considera-se a recorrência simplificada

$$T(n) = 2\,T(n/2) + n$$

definida apenas para n potência de 2.

Prova-se por indução em n que $T(n) = n + n \lg n = \mathrm{O}(n \lg n)$.

Prova

Afirmação: A recorrência

$$T(n) = \begin{cases} 1 & \text{se } n = 1 \\ 2\,T(n/2) + n & \text{se } n \geq 2,\ n \text{ potência de } 2 \end{cases}$$

tem como soluçao $T(n) = n + n \lg n$.

Prova

Afirmação: A recorrência

$$T(n) = \begin{cases} 1 & \text{se } n = 1 \\ 2\,T(n/2) + n & \text{se } n \geq 2,\ n \text{ potência de } 2 \end{cases}$$

tem como soluçao $T(n) = n + n \lg n$.

Prova: Por indução em n.

Base: $n = 1$

$$T(1) = 1 = 1 + 1 \cdot 0 = 1 + 1 \lg 1.$$

Prova

Afirmação: A recorrência

$$T(n) = \begin{cases} 1 & \text{se } n = 1 \\ 2\,T(n/2) + n & \text{se } n \geq 2\text{, } n \text{ potência de } 2 \end{cases}$$

tem como soluçao $T(n) = n + n \lg n$.

Prova: Por indução em n.

Base: $n = 1$

$$T(1) = 1 = 1 + 1 \cdot 0 = 1 + 1 \lg 1.$$

Passo: $n \geq 2$

$$\begin{aligned} T(n) &= 2\,T(n/2) + n \\ &= 2\,(n/2 + (n/2)\lg(n/2)) + n \qquad \text{por indução} \\ &= 2n + n\lg(n/2) \\ &= 2n + n(\lg n - 1) \\ &= n + n \lg n. \end{aligned}$$

Resolução de recorrências

Mas como descobrimos que $T(n) = n + n \lg n$?

Resolução de recorrências

Mas como descobrimos que $T(n) = n + n \lg n$? No chute!

Resolução de recorrências

Mas como descobrimos que $T(n) = n + n \lg n$? No chute!

Uma maneira de se obter um "chute" de solução de recorrência é desenrolando a recorrência.

Resolução de recorrências

Mas como descobrimos que $T(n) = n + n \lg n$? No chute!

Uma maneira de se obter um “chute” de solução de recorrência é desenrolando a recorrência.

$$
\begin{aligned}
T(n) &= 2\,T(n/2) + n \\
&= 2\,(2\,T(n/2^2) + n/2) + n \;=\; 2^2 T(n/2^2) + 2n \\
&= 2^2\,(2\,T(n/2^3) + n/2^2) + 2n \;=\; 2^3 T(n/2^3) + 3n \\
&= 2^3\,(2\,T(n/2^4) + n/2^3) + 3n \;=\; 2^4 T(n/2^4) + 4n \\
&= \cdots \\
&= 2^k\,T(n/2^k) + kn,
\end{aligned}
$$

onde $k = \lg n$. Disso concluímos que

$$T(n) = n + n \lg n.$$

Exercícios

Exercício 2.A
Interprete e prove a afirmação $\mathrm{O}(n^2) + \mathrm{O}(n^2) + \mathrm{O}(n^2) = \mathrm{O}(3n^2)$.

Exercício 2.B
Interprete e prove a afirmação $n\mathrm{O}(n) = \mathrm{O}(n^2)$.

Exercício 2.C
Interprete e prove a afirmação $\mathrm{O}(3n^2 + 4n) = \mathrm{O}(n^2)$.

Exercício 2.D (propriedade transitiva)
Suponha $T(n) = \mathrm{O}(f(n))$ e $f(n) = \mathrm{O}(g(n))$.
Mostre que $T(n) = \mathrm{O}(g(n))$.
Dê um exemplo interessante.

Exercício 2.E (regra da soma, caso especial)
Suponha que $T(n) = \mathrm{O}(f(n))$ e mostre que $T(n) + f(n) = \mathrm{O}(f(n))$.
Dê um exemplo interessante.

Exercício 2.E' (regra da soma, geral)
Suponha $T_1(n) = \mathrm{O}(f_1(n))$ e $T_2(n) = \mathrm{O}(f_2(n))$. Se $f_1(n) = \mathrm{O}(f_2(n))$, mostre que $T_1(n) + T_2(n) = \mathrm{O}(f_2(n))$.

Mais exercícios

Exercício 2.F

O que significa "$T(n) = n^2 + \mathrm{O}(n)$"?
Mostre que se $T(n) = n^2 + \mathrm{O}(n)$ então $T(n) = \mathrm{O}(n^2)$.

Exercício 2.G

O que significa "$T(n) = n\mathrm{O}(\lg n)$"? Mostre que $T(n) = n\mathrm{O}(\lg n)$ se e só se $T(n) = \mathrm{O}(n \lg n)$.

Exercício 2.H

Interprete e prove a afirmação $7 \cdot \mathrm{O}(n) = \mathrm{O}(n)$.

Exercício 2.I

Interprete e prove a afirmação $\mathrm{O}(n) + \mathrm{O}(n) = \mathrm{O}(n)$.

Exercício 2.J

Prove que $\mathrm{O}(n) = \mathrm{O}(n^2)$. É verdade que $\mathrm{O}(n^2) = \mathrm{O}(n)$?

Exercício 2.K

Interprete e prove a afirmação $(n + 2) \cdot \mathrm{O}(1) = \mathrm{O}(n)$.

Mais exercícios ainda

Exercício 2.L

Interprete e prove a afirmação $\underbrace{\mathrm{O}(1) + \cdots + \mathrm{O}(1)}_{n+2} = \mathrm{O}(n)$.

Exercício 2.M

Prove que $\mathrm{O}(1) + \mathrm{O}(1) + \mathrm{O}(1) = \mathrm{O}(1)$.

É verdade que $\mathrm{O}(1) = \mathrm{O}(1) + \mathrm{O}(1) + \mathrm{O}(1)$?

Exercício 2.N

Interprete e prove a afirmação $\mathrm{O}(f) + \mathrm{O}(g) = \mathrm{O}(f + g)$.

Exercício 2.O

Prove que $n^2 + 10n + 20 = \Omega(n^2)$. Prove que $n^2 - 10n - 20 = \Theta(n^2)$.

Exercício 2.P

Prove que $n = \Omega(\lg n)$.

Exercício 2.Q

Prove que $\lg n = \Theta(\log_{10} n)$.

Exercício 2.R

É verdade que $2^n = \Omega(3^n)$?

Mais exercícios

Exercício 2.S

É verdade que $2n^3 + 5\sqrt{n} = \Theta(n^3)$?

Exercício 2.T

Suponha que os algoritmos $\mathcal{A}$ e $\mathcal{B}$ só dependem de um parâmetro n. Suponha ainda que $\mathcal{A}$ consome $S(n)$ unidades de tempo enquanto $\mathcal{B}$ consome $T(n)$ unidades de tempo. Quero provar que algoritmo $\mathcal{A}$ é pelo menos tão eficiente quanto o algoritmo $\mathcal{B}$ (no sentido assintótico). Devo mostrar que existe $f(n)$ tal que

$S(n) = \mathrm{O}(f(n))$ e $T(n) = \mathrm{O}(f(n))$?

$S(n) = \mathrm{O}(f(n))$ e $T(n) = \Omega(f(n))$?

$S(n) = \Omega(f(n))$ e $T(n) = \mathrm{O}(f(n))$?

$S(n) = \Omega(f(n))$ e $T(n)\Omega(f(n))$?

Que devo fazer para mostrar que $\mathcal{A}$ é mais eficiente que $\mathcal{B}$?

Exercício 2.U

Mostre que o consumo de tempo do algoritmo INTERCALA é $\Theta(n)$, sendo n o número de elementos do vetor que o algoritmo recebe.