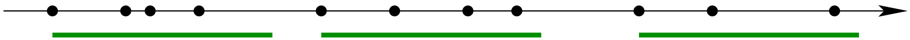
Exercício 1

(CLRS 16.2-5) Descreva um algoritmo eficiente que dado um conjunto $\{x_1, \ldots, x_n\}$ de pontos ordenados na reta real, determina uma coleção mínima de intervalos fechados unitários (tamanho 1) que contém todos os pontos (por mínima, quero dizer menor número de intervalos).



Observação. Se os pontos não estiverem ordenados, é necessário um passo adicional de ordenação com custo $O(n \lg n)$.

Exercício 1 – Solução

Escolha gulosa:

Exercício 1 – Solução

Escolha gulosa:

- Seja $I = [x_1, x_1 + 1]$.

Exercício 1 – Solução

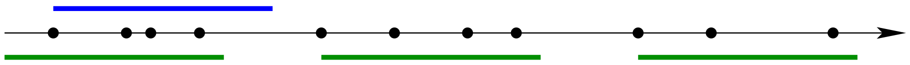
Escolha gulosa:

- Seja $I = [x_1, x_1 + 1]$.
- Mostraremos que existe uma solução ótima que contém I.

Exercício 1 – Solução

Escolha gulosa:

- Seja $I = [x_1, x_1 + 1]$.
- Mostraremos que existe uma solução ótima que contém I.
- Seja $\mathcal{I}$ uma solução ótima.

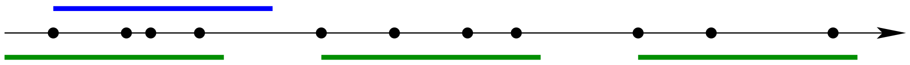


Exercício 1 – Solução

Escolha gulosa:

- Seja $I = [x_1, x_1 + 1]$.
- Mostraremos que existe uma solução ótima que contém I.
- Seja $\mathcal{I}$ uma solução ótima.

  Se $I \in \mathcal{I}$, então nada há a provar.



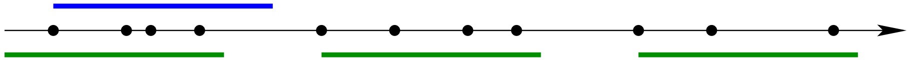
Exercício 1 – Solução

Escolha gulosa:

- Seja $I = [x_1, x_1 + 1]$.
- Mostraremos que existe uma solução ótima que contém I.
- Seja $\mathcal{I}$ uma solução ótima.

  Se $I \in \mathcal{I}$, então nada há a provar.

  Senão, seja I' o intervalo em $\mathcal{I}$ que contém x_1. Como x_1 é o menor ponto, a coleção $\mathcal{I}' = \mathcal{I} - \{I'\} \cup I$ também cobre todos os pontos. Portanto, $\mathcal{I}'$ é uma solução ótima que contém I.



Subestrutura ótima:

Exercício 1 – Solução

Subestrutura ótima:

- Seja $\mathcal{I}$ uma solução ótima que contém $I = [x_1, x_1 + 1]$.

Exercício 1 – Solução

Subestrutura ótima:

- Seja $\mathcal{I}$ uma solução ótima que contém $I = [x_1, x_1 + 1]$.
- Claramente $\mathcal{I}' = \mathcal{I} - \{I\}$ é uma solução ótima do subproblema X' formado pelos pontos x_i em $(x_1 + 1, x_n]$.

Exercício 1 – Solução

Subestrutura ótima:

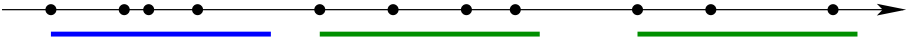
- Seja $\mathcal{I}$ uma solução ótima que contém $I = [x_1, x_1 + 1]$.
- Claramente $\mathcal{I}' = \mathcal{I} - \{I\}$ é uma solução ótima do subproblema X' formado pelos pontos x_i em $(x_1 + 1, x_n]$.
- Caso contrário, existiria uma solução $\mathcal{I}''$ desta instância com menos intervalos.

Exercício 1 – Solução

Subestrutura ótima:

- Seja $\mathcal{I}$ uma solução ótima que contém $I = [x_1, x_1 + 1]$.
- Claramente $\mathcal{I}' = \mathcal{I} - \{I\}$ é uma solução ótima do subproblema X' formado pelos pontos x_i em $(x_1 + 1, x_n]$.
- Caso contrário, existiria uma solução $\mathcal{I}''$ desta instância com menos intervalos.

  Mas então $\mathcal{I}'' \cup \{I\}$ seria uma solução da instância original com menos intervalos que $\mathcal{I}$, uma contradição.



Exercício 1 – Algoritmo recursivo

$\textsc{Intervalos}(X, i, n)$
Entrada: pontos $x_i, \ldots, x_n$ (ordenados).
Saída: coleção mínima de intervalos unitários que cobre os pontos.

se $i > n$ **então devolva** $\emptyset$
$j \leftarrow i + 1$
enquanto $j \leq n$ e $x_j \leq x_i + 1$ **faça**
    $j \leftarrow j + 1$
devolva $\{[x_i, x_i + 1]\} \cup \textsc{Intervalos}(X, j, n)$

Chamada inicial: $\textsc{Intervalos}(X, 1, n)$

Complexidade: $O(n)$

Exercício 1 – Algoritmo iterativo

$\textsc{Intervalos}(X, n)$
Entrada: pontos $X = \{x_1, \ldots, x_n\}$ (ordenados).
Saída: coleção mínima de intervalos unitários que cobre os pontos.

$\mathcal{I} \leftarrow \{[x_1, x_1 + 1]\}$, $i \leftarrow 1$, $j \leftarrow 2$
enquanto $j \leq n$ **faça**
    se $x_j > x_i + 1$
        então
            $\mathcal{I} \leftarrow \mathcal{I} \cup \{[x_j, x_j + 1]\}$
            $i \leftarrow j$
    $j \leftarrow j + 1$
devolva $\mathcal{I}$

Complexidade: $O(n)$

Exercício 2

(CLRS 16.2-7) Suponha que A e B sejam vetores de números reais positivos de tamanho n. Queremos rerranjar a ordem dos elementos de A e B de modo a maximizar o **payoff (lucro)**

$$\prod_{i=1}^{n} A[i]^{B[i]}.$$

Descreva um algoritmo eficiente para resolver este problema.

Exercício 2

(CLRS 16.2-7) Suponha que A e B sejam vetores de números reais positivos de tamanho n. Queremos rerranjar a ordem dos elementos de A e B de modo a maximizar o **payoff (lucro)**

$$\prod_{i=1}^{n} A[i]^{B[i]}.$$

Descreva um algoritmo eficiente para resolver este problema.

Observação. Outra maneira de pensar é que queremos emparelhar cada elemento de A com exatamente um elemento de B de modo a maximizar o valor do payoff.

Exercício 2 – Solução

Escolha gulosa:

Exercício 2 – Solução

Escolha gulosa:

- Sejam a o maior elemento de A e b o maior elemento de B.

Exercício 2 – Solução

Escolha gulosa:

- Sejam a o maior elemento de A e b o maior elemento de B.
- Mostraremos que existe uma solução ótima na qual a é emparelhado com b.

Exercício 2 – Solução

Escolha gulosa:

- Sejam a o maior elemento de A e b o maior elemento de B.
- Mostraremos que existe uma solução ótima na qual a é emparelhado com b.
- Considere uma permutação ótima A', B' de A, B que maximiza o payoff.

Exercício 2 – Solução

Escolha gulosa:

- Sejam a o maior elemento de A e b o maior elemento de B.
- Mostraremos que existe uma solução ótima na qual a é emparelhado com b.
- Considere uma permutação ótima A', B' de A, B que maximiza o payoff.

  Se a e b estão emparelhados nesta solução, então nada há a fazer.

Exercício 2 – Solução

Escolha gulosa:

- Sejam a o maior elemento de A e b o maior elemento de B.
- Mostraremos que existe uma solução ótima na qual a é emparelhado com b.
- Considere uma permutação ótima A', B' de A, B que maximiza o payoff.

  Se a e b estão emparelhados nesta solução, então nada há a fazer.

  Suponha então que a esteja emparelhado com y e que x esteja emparelhado com b.

	1	2	3	$\cdots$	$n-1$	n
A'	a	x				
B'	y	b				

Exercício 2 – Solução

Escolha gulosa:

- Considere uma nova permutação A'', B'' em que trocamos as posições de y e b e mantemos a ordem dos demais elementos.

	1	2	3	$\cdots$	$n-1$	n
A'	a	x				
B'	y	b				

	1	2	3	$\cdots$	$n-1$	n
A''	a	x				
B''	b	y				

Exercício 2 – Solução

Escolha gulosa:

- Considere uma nova permutação A'', B'' em que trocamos as posições de y e b e mantemos a ordem dos demais elementos.
- Sejam P' o payoff ótimo e P'' o payoff de A'', B''. Então

$$P'' = P' \cdot \frac{1}{a^y} \cdot \frac{1}{x^b} \cdot a^b \cdot x^y.$$

	1	2	3	$\cdots$	$n-1$	n
A'	a	x				
B'	y	b				

	1	2	3	$\cdots$	$n-1$	n
A''	a	x				
B''	b	y				

Exercício 2 – Solução

Escolha gulosa:

- Considere uma nova permutação A'', B'' em que trocamos as posições de y e b e mantemos a ordem dos demais elementos.
- Sejam P' o payoff ótimo e P'' o payoff de A'', B''. Então

$$P'' = P' \cdot \frac{1}{a^y} \cdot \frac{1}{x^b} \cdot a^b \cdot x^y.$$

Como

$$\frac{a^b}{a^y} \cdot \frac{x^y}{x^b} = a^{b-y} \cdot x^{y-b} = \left(\frac{a}{x}\right)^{b-y} \geq 1,$$

A'', B'' é uma solução ótima na qual a e b são emparelhados.

	1	2	3	$\cdots$	$n-1$	n
A'	a	x				
B'	y	b				

	1	2	3	$\cdots$	$n-1$	n
A''	a	x				
B''	b	y				

Exercício 2 – Solução

Subestrtutura ótima:

Exercício 2 – Solução

Subestrtutura ótima:

- Seja A', B' uma solução ótima. Sem perda de generalidade, suponha que $A'[1] = a$ e $B'[1] = b$.

	1	2	3	$\cdots$	$n-1$	n
A'	a					
B'	b					

Exercício 2 – Solução

Subestrtutura ótima:

- Seja A', B' uma solução ótima. Sem perda de generalidade, suponha que $A'[1] = a$ e $B'[1] = b$.
- Então $A'[2..n], B'[2..n]$ é uma solução ótima de $A[2..n], B[2..,n]$.

	1	2	3	$\cdots$	$n-1$	n
A'	a					
B'	b					

Exercício 2 – Solução

Subestrtutura ótima:

- Seja A', B' uma solução ótima. Sem perda de generalidade, suponha que $A'[1] = a$ e $B'[1] = b$.
- Então $A'[2 \,.\,.\, n], B'[2 \,.\,.\, n]$ é uma solução ótima de $A[2 \,.\,.\, n], B[2 \,.\,.\,, n]$.
- Caso contrário, existiria uma solução $A''[2 \,.\,.\, n], B''[2 \,.\,.\, n]$ com payoff maior que $A[2 \,.\,.\, n], B[2 \,.\,.\, n]$.

	1	2	3	$\cdots$	$n-1$	n
A'	a					
B'	b					

	2	3	$\cdots$	$n-1$	n
A''					
B''					

Subestrtutura ótima:

- Seja A', B' uma solução ótima. Sem perda de generalidade, suponha que $A'[1] = a$ e $B'[1] = b$.
- Então $A'[2\,..\,n], B'[2\,..\,n]$ é uma solução ótima de $A[2\,..\,n], B[2\,..\,,n]$.
- Caso contrário, existiria uma solução $A''[2\,..\,n], B''[2\,..\,n]$ com payoff maior que $A[2\,..\,n], B[2\,..\,n]$.

  Mas então setando $A''[1] = a$ e $B''[1] = b$ obtemos uma solução de A, B com payoff maior que A', B', uma contradição.

	1	2	3	$\cdots$	$n-1$	n
A'	a					
B'	b					

	2	3	$\cdots$	$n-1$	n
A''					
B''					

Exercício 2 – Algoritmo iterativo

$\textsc{Payoff}(A, B, n)$
Entrada: vetores de inteiros positivos de tamanho n.
Saída: permutações de A e B que maximizam o payoff.
1. Ordene A
2. Ordene B
3. **devolva** A, B

Complexidade: $O(n \lg n)$

Exercício 2 – Algoritmo iterativo

$\text{PAYOFF}(A, B, n)$

Entrada: vetores de inteiros positivos de tamanho n.

Saída: permutações de A e B que maximizam o payoff.

1. Ordene A
2. Ordene B
3. **devolva** A, B

Complexidade: $O(n \lg n)$

Observação. A versão recursiva teria que usar duas filas de prioridade com chaves em A e B para encontrar os máximos em cada chamada recursiva. A complexidade seria a mesma, se usarmos um maxheap (do Heapsort).