

Leonardo Rodrigues Marques

RA: 178610

Leonardo Rodrigues Marques

Corrigir a questão 02.

1a) 2,0

① Para mostrar que o problema tem estrutura ótima, temos que mostrar que existem ótimas soluções para subproblemas. Portanto, dada duas strings de tamanhos  $n, m$  e uma string intercalada  $Z = Y + X$  de tamanho  $n+m$ . Para verificarmos se  $X \neq Y$  estes contidos em  $Z$ , basta comparar elementos  $X_m$  ou  $Y_m$  com  $Z_{n+m}$ . Dependendo do resultado, obtemos strings  $X_{n-s}$  ou  $Y_{m-s}$  e  $Z_{n+m-s}$ , que são subproblemas de mesmo malhação e admitem a mesma solução ótima. Nesse logíco sucessiva, podemos formar uma relação de recorrências:

$$S[i, j, k] = \begin{cases} \text{false} & , i=0 \text{ e } Y_j \neq Z_k \quad (1) \\ \text{false} & , j=0 \text{ e } X_i \neq Z_k \quad (2) \\ \text{true} & , i, j, k = 0, 0, 0 \quad (3) \\ S[i-1, j, k-1] & , Z_k = X_i \quad (4) \\ S[i, j-1, k-1] & , Z_k = Y_j \quad (5) \end{cases}$$

e se  $Z_k \neq X_i$  e  $Z_k \neq Y_j$ ?

(1)  $\Rightarrow$  conduz em que já percorremos todo o vetor  $X$  e ao compararmos elemento de  $Y$  com  $Z$ , são diferentes, portanto não temos uma intercalação.

(2)  $\Rightarrow$  conduz em que já percorremos todo o vetor  $Y$  e ao compararmos elemento de  $X$  com  $Z$ , são diferentes, portanto não temos uma intercalação.

(3)  $\Rightarrow$  se os índices  $i, j, k$  são 0 dos vetores  $X, Y, Z$ , ento percorremos os vetores e constatou-se que há uma intercalação (~~errado~~) (todos os elementos batem).

(4)  $\Rightarrow$  Se o elemento  $X_i = Z_k$ , então existe um elemento de  $x$  em  $z$  e portanto podemos verificar para instâncias  $i-1, k-1$  menores.

\*out

(5)  $\Rightarrow$  Se o elemento  $Y_j = Z_k$ , então existe um elemento de  $y$  em  $z$  e portanto podemos verificar para instâncias  $j-1, k-1$  menores.

1b) 0,5

esta Intercalado ( $Z, X, Y, i, j, k, H$ ) {

se  $i, j, k = 0$ , então: (3)

| devolve (true)

| senão

| devolve (false)

Obs:  $H$  é uma matriz nxm com valores false.

$i, j, k$  são índices do último elemento de  $X, Y, Z$ .

Seu algoritmo vai retornar false sempre que a  $i > 0$  ou  $j > 0$  ou  $k > 0$

se  $H[i, j] \neq H[m, m]$  ?

| devolve  $H[i, j]$

se  $i=0$  e  $Y_j \neq Z_k$ , então (1)

|  $H[i, j] = \text{false}$

| devolve (false)

senão:

|  $H[i, j] = \text{true}$

O resultado depende do retorno

| devolve (estaIntercalado ( $Z, X, Y, i, j-1, k, H$ ))<sup>(5)</sup>

se  $j=0$  e  $X_i \neq Z_k$ , então (2)

|  $H[i, j] = \text{false}$

| devolve (false)

senão

|  $H[i, j] = \text{true}$

| devolve (estaIntercalado ( $Z, X, Y, i-1, j, k, H$ ))<sup>(4)</sup>

se  $Z_k = X_i$  então:

|  $x = \text{esta Intercalado} (Z, X, Y, i-1, j, k-1, H)$  <sup>(4)</sup>

se  $Z_k = Y_j$  então:

|  $y = \text{esta Intercalado} (Z, X, Y, i, j-1, k-1, H)$  <sup>(5)</sup>

|  $H[i, j] = x *_{\text{out}} y$

| devolve ( $x$  ou  $y$ )

Foi utilizada uma matriz  $H$  de tamanho  $n \times m$  para armazenar os valores true ou false, evitando desperdício com resultados de recursão. Foi baseado no tratativa do subproblema máximo comum, portanto, provavelmente<sup>?</sup> a complexidade de tempo e espaço também deve ser na ordem  $O(nm)$ . 1c) 0,5

②

2a) 2,0

2b) 1,5

2c) 1,0

Para mostrar que o problema possui subestrutura ótima, temos que mostrar que existem soluções ótimas para subproblemas. Se examinarmos o deslocamento menor custoso de  $A_1$  para  $A_n$ , necessariamente envolvemos os subcaminhos de  $A_1$  até  $A_n$ . Nesses subcaminhos, temos que verificar quais os menores custos (soluções ótimas), retornando a natureza do problema. A partir disso, podemos sintetizar uma fórmula de recorrência:

Seja  $V[k]$ , o custo até a aldeia  $A_k$  e  $T$ , a tabela contendo os custos  $t_{ik}$ .

$$V[k] = \begin{cases} 0 & , k=1 \\ \min_{1 \leq i < k} \{ \cancel{oo}, V[i] + t_{1,k} \} , \cancel{\forall t_{1,k} \in T} \} & , k \neq 1 \end{cases}$$

A base da relação é o estado inicial onde se encontra. Como o estado inicial se encontra em  $A_1$ , logicamente o custo nessa posição é 0. A partir de  $A_1$ , os deslocamentos para  $A_2$  até  $A_n$  possuem custos  $t_{1,k} > 0$ .

Portanto, devemos procurar pelo menor custo para todos os subcaminhos até uma aldeia  $A_k$ . A partir desse processo de minimização, conseguimos construir um caminho final até  $A_n$  de menor custo.

O pseudo-algoritmo em questão é:

min-custo ( $T, C, n$ ):

para $C[1]:=0$ para $i:=2$ até $n$ faça $C[i]=\infty$ para $i:=1$ até $n$ faça: para $j:=i+1$ até $n$ faça: se $C[j] > C[i] + T[i,j]$ então: $C[j] = C[i] + T[i,j]$ .	$O(1)$ $O(n-1)$ $O(n)$ $O(n-1)$ $O(1)$ $O(1)$
--	--

de volta  $C[n]$   $O(1)$

Tempo =  $O(n^2)$   
 Espaço =  $O(n)$ .

$$O(1) + O(n-1) + O(n)(O(n-1) + O(1) + O(1)) = O(n) + O(n)O(n) = O(n^2)$$

③ O algoritmo de questão 02 procura o menor custo de aldeias  $A_2$  até  $A_n$ . Durante o processo de manutenção, usamos o custo do deslocamento da aldeia  $A_{k+1}$  para  $A_k$ , e descartamos a informação da aldeia de origem do deslocamento (que pode ser configurada como dístrito ~~segundo~~ instâncias). Por exemplo: 3) Não corrigido.

$V_1$	$V_2$	$V_3$
0	2	4

$$V[2] = \min \{ \quad \quad \quad V[3] = \min \{ \quad \quad \quad \}$$

$$t_{2,2} = 2 \quad t_{2,3} = 4$$

$$\{ = 2 \quad V[2] + t_{2,3} = 2 + 3 = 5$$

$$\{ = 4$$

Mesmo abstracionando, obtemos apenas o custo e descartamos as aldeias de origem.

Portanto, se além dos custos, pudermos armazenar anterior  $A_k$ , e posteriormente recuperarmos o caminho entre as aldeias.

	1	2	3
V	0	2	
A	1.	1	

$$V[2], A[2] = \min \{ \quad \quad \quad V[3], A[3] = \min \{ \quad \quad \quad \}$$

$$t_{2,2} = 2^*$$

$$t_{2,3} = 4^*$$

$$V[2] + t_{2,3} = 2 + 3 = 5$$

$$\{ = 2, 2$$

$$\{ = 4, 1$$

Automaticamente, todos os instâncias maiores até  $A_n$  são abrangidas.

Nessa forma, temos que o elemento  $A[n]$  possui referências para  $A[k]$ , bem com menor custo e sucessivamente até chegarmos na origem (base)  $A_1$ . Por fim, temos o percurso onde é possível informar os lugares onde devolver as canoas.

menor-custo - caminho ( $T, C, A, n$ ):

:

$$\left. \begin{array}{l} \text{se } C[j] > C[i] + T_{i,j} \text{ então} \\ \quad C[j] = C[i] + T_{i,j} \\ \quad A[j] = i \end{array} \right\} \begin{array}{l} * \text{referência} \\ \text{a Questão 2} \end{array}$$

: