D S T Q Q S S
D L M M J V S

Tiago de Paula Alves R.A.: 187679

Tiago de Paula Alves

CORRIGIR A QUESTÃO 1

1. a) Considere a operação "$\oplus$" como a concatenação de cadeia e "X_i" como a notação de prefixo adotada em aula.

Teorema (subestrutura ótima). Sejam m e n inteiros positivos e considere $X = \langle x_1, \ldots, x_m \rangle$, $Y = \langle y_1, \ldots, y_n \rangle$ e $Z = \langle z_1, \ldots, z_{n+m} \rangle$ como cadeias de caracteres de Σ. Então, Z é uma intercalação de X e Y se e somente se ~~uma~~ pelo menos uma das condições a seguir for atendida:

1. $x_m = z_{n+m}$ e Z_{m+n-1} é intercalação de X_{m-1} e Y;
2. $y_n = z_{n+m}$ e Z_{m+n-1} é intercalação de X e Y_{n-1}.

Demonstração. Considere $Z' = Z_{m+n-1}$, $X' = X_{m-1}$ e $Y' = Y_{n-1}$.

($\rightarrow$) Suponha que Z é uma intercalação de X e Y e que a condição 1 não foi atendida. Então temos os seguintes casos:

Caso 1: $x_m \neq z_{m+n}$. Assim, se $y_n \neq z_{m+n}$, então Z não poderia ser uma intercalação de X e Y, logo, $y_n = z_{m+n}$. Suponha agora que Z' não é uma intercalação de X e Y', ou seja, não é possível decompor Z' em subcadeias disjuntas iguais a X e Y'.

Para que exista uma subcadeia S de $Z = Z' \oplus \langle z_{m+n} \rangle$ tal que $S = Y' \oplus \langle y_n \rangle = Y$, temos que z_{m+n} deverá ser parte de S. Se X não é subcadeia de Z', como z_{m+n} seria parte da subcadeia igual a Y, então X também não seria subcadeia de Z. Por outro lado, se Y' não é subcadeia de Z', então $Y = Y' \oplus \langle y_n \rangle$ também não poderá ser subcadeia de Z.

Portanto, temos que Z' deve ser uma intercalação de X e Y', ou seja, a condição 2 deve ser atendida.

Caso 2: $x_m = z_{m+n}$ e Z' não é intercalação de X' e Y. Suponha que ~~z_{m+m}~~ z_{m+n} é parte da subcadeia S de Z que é igual a X. No entanto, como Z' não é intercalação de X' e Y, temos que $Z = Z' \oplus \langle z_{m+n} \rangle$ não pode ser intercalação de $X = X' \oplus \langle x_m \rangle = S$ e Y. Portanto, z_{m+n} deve ser parte de Y.

Como z_{m+n} é o último elemento de Z, então $y_n = z_{m+n}$. Além disso, assim como no caso anterior, temos que z_{m+n} é parte da subcadeia igual a Y, então Z' deve ser uma intercalação de X e Y'.

Por fim, como os casos são exaustivos, temos que se a condição 1 é falsa, a condição 2 deve ser verdade. Logo, para que Z seja a intercalação proposta, uma das condições deve ser atendida.

($\leftarrow$) Considere agora que a condição 1 é verdade. Como $Z = Z' \oplus \langle z_{m+n} \rangle$, $X = X' \oplus \langle x_m \rangle$ e $z_{m+n} = x_m$, então as operações de concatenação matêm a propriedade de intercalação, nesse caso. Portanto, Z é uma intercalação de X e Y.

A demonstração a partir da condição 2 é similar. ∎

Além do caso geral discutido acima, é possível que X ou Y seja vazia ($\emptyset$), isto é, $m = 0$ ou $n = 0$. O caso base acontece quando ambas são vazias. A partir da subestrutura ótima e dos casos com cadeias vazias, podemos ver que a recorrência para verificação de intercalação será dada por:

1. $\emptyset$ é intercalação de $\emptyset$ e $\emptyset$
2. Z é intercalação de X e $\emptyset$ $\leftrightarrow$ $z_m = x_m$ e Z_{m-1} é intercalação de X_{m-1} e $\emptyset$
3. Z é intercalação de $\emptyset$ e Y $\leftrightarrow$ $z_n = y_n$ e Z_{n-1} é intercalação de $\emptyset$ e Y_{n-1}
4. Z é intercalação de X e Y $\leftrightarrow$
   - $z_{m+n} = x_m$ e Z_{m+n-1} é intercalação de X_{m-1} e Y; ou
   - $z_{m+n} = y_n$ e Z_{m+n-1} é intercalação de X e Y_{n-1}

① b) Considere VERDADEIRO como 1, FALSO como 0 e NULO como uma constante diferente de 1 e de 0.

```
INTERCALACAO(X, m, Y, n, Z)
    Seja I[0..m][0..n] uma matriz

    para i = 0 até m
        para j = 0 até n
            I[i][j] ← NULO

    retorna INTERCALACAO-REC(X, m, Y, n, Z, I)
```

```
INTERCALACAO-REC(X, m, Y, n, Z, I)
  se I[m][n] ≠ NULO
      retorna I[m][n]

  se m == 0 e n == 0
      I[0][0] ← VERDADEIRO
  senão
      I[m][n] ← FALSO
      se m ≠ 0 e X[m] == Z[m+n]
          I[m][n] ← I[m][n] ou
                    INTERCALACAO-REC(X, m-1, Y, n, Z, I)
      se n ≠ 0 e Y[n] == Z[m+n]
          I[m][n] ← I[m][n] ou
                    INTERCALACAO-REC(X, m, Y, n-1, Z, I)

  retorna I[m][n]
```

① c) Considerando m' e n' as entradas das chamadas recursivas de INTERCALA-REC, é ~~fácil~~ possível notar que $0 \leq m' \leq m$ e $0 \leq n' \leq n$. Além disso, cada nível da recursão faz no máximo duas novas chamadas, ou seja, o número de chamadas não cresce com o valor de m o n.

Assim, a partir do momento que a matriz de memorização I está preenchida, qualquer chamada leva tempo constante. Portanto, a complexidade da função recursiva é limitada pelo tamanho da matriz, ou seja, é $O((m+1)(n+1)) = O(mn)$.

Então, a função INTERCALACAO, que inicializa a tabela e a recursão, tem tempo:

$$T(m,n) = \sum_{i=0}^{m} \sum_{j=0}^{n} \Theta(1) + O(mn)$$

$$= \Theta((m+1)(n+1)) + O(mn)$$

$$= \Theta(mn)$$

Para a complexidade de espaço, podemos ver que a função recursiva usa espaço constante em cada chamada. Como as chamadas são limitadas pelo tamanho da tabela de memorização, então o espaço utilizado é $O(mn)$. Para a função final, a criação da tabela I, com dimensões $(m+1) \times (n+1)$, também determina a complexidade. Portanto, $E(m,n) = \Theta(mn)$.

② a)

Teorema (subestrutura ótima). Seja P um percurso da aldeia A_1 até A_n com custo mínimo. Então, todo subpercurso $R \subseteq P$ que termina em A_n também é mínimo.

Demonstração. Suponha um percurso P com custo $c(P)$ mínimo de A_1 até A_n e uma aldeia $A_{1 \leq k \leq n}$ como parte do percurso. Logo, temos o subpercurso $R \subseteq P$ de A_k até A_n. Suponha também um percurso R' de A_k a A_n com custo $c(R')$ mínimo.

Assim, se $c(R') < c(R)$, então podemos formar o percurso total $P' = (P \setminus R) \cup R'$, tal que $c(P') = c(P) - c(R) + c(R') < c(P)$. Ou seja, P não poderia ser considerado mínimo.

Logo, $c(R) \leq c(R')$, isto é, R deve ser mínimo. Além disso, como A_k era arbitrário, todo subpercurso de P também deve ter custo mínimo. ■

Considere que C_i é o menor custo de um percurso da aldeia A_i para a A_n, sendo $1 \leq i \leq n$. Como os subpercursos são ótimos, podemos considerar todas as possíveis paradas $A_{i < j \leq n}$ com seus percursos ótimos até A_n, tomando o menor deles como parte do percurso ótimo de A_i. Note que se $i = n$, então não resta nenhuma aldeia no percurso, ou seja, $C_i = 0$.

Assim,

$$C_i = \min_{i<j\leq n}\{t_{i,j} + C_j\}$$

$$C_n = 0$$

Nessa relação, o custo ótimo C_i depende apenas dos custos C_j das aldeias seguintes, com $i<j$, já que $t_{i,j} > 0$ e não é possível alugar uma canoa de $A_{j>i}$ para A_i. Assim, podemos calcular os custos a partir da última aldeia A_n, sem necessidade de recursão ou memorização.

---

2 b)

```
Custo-Mínimo(t, n)
Seja C[1.. n-1] um novo vetor.

para i = n-1 descendo até 1
    cmin ← t[i][n]
    para j = i+1 até n-1
        custo ← t[i][j] + C[j]
        cmin ← min(cmin, custo)
    C[i] ← cmin

retorna C[1]
```

② c) Vamos considerar que as linhas 1 e 10 executam em tempo constante a_1, que a 4 e a 8 executam em a_2 e que 6 e 7 são ~~tempo~~ em tempo a_3. Assim, podemos descrever o tempo de execução do algoritmo por:

$$T(n) = a_1 + \sum_{i=1}^{n-1} \left(a_2 + \sum_{j=i+1}^{n-1} a_3 \right)$$

$$= a_1 + \sum_{i=1}^{n-1} a_2 + a_3 \sum_{i=1}^{n-1} n - a_3 \sum_{i=1}^{n-1} i - a_3 \sum_{i=1}^{n-1} 1$$

$$= a_1 + a_2(n-1) + a_3 n(n-1) - a_3 \frac{n(n-1)}{2} - a_3(n-1)$$

$$= \frac{a_3}{2} n^2 + \frac{2a_2 - 3a_3}{2} n + a_1 - a_2 + a_3$$

Ou seja, $T(n) \in \Theta(n^2)$, como requerido. Além disso, o único espaço adicional é do vetor C de tamanho $n-1$. Então, a complexidade de ~~espaç~~ espaço será dada por:

$$E(n) = n - 1 + \Theta(1) = \Theta(n)$$

(3) Para conseguir a sequência de aldeias, podemos guardar os índices que minimizam o cálculo do custo, dado pela recorrência do item 2a). Assim, teremos um vetor P de tamanho n onde $C_i = t_{i,P_i} + C_{P_i}$, ou seja, P_i é o índice da próxima aldeia no caminho ótimo partindo de A_i.

Na prática, esse índice poderia ser acumulado em uma variável pmin, assim como é feito com cmin (linha 4), sendo trocado sempre que cmin fosse atualizado (linha 7). Ou seja, teríamos uma nova linha com "$pmin \leftarrow j$" se custo < cmin, mantendo a nova invariante de que

$$cmin = \min_{i<k\leq j} \{t_{i,k} + C_k\} = t_{i,pmin} + C_{pmin}$$

Ao final do laço da linha 5, teremos então que $P_i = pmin$, além de $C_i = cmin$, como já acontecia na linha 8.

Por fim, com esse vetor P, podemos montar o caminho ótimo M partindo de A_1. Assim, teremos $M_1 = P_1$ e $M_{i+1} = P_{M_i}$, até encontrar um valor final $1 \leq f \leq n$ em que $M_f = n$, ou seja, a última aldeia. Esse caminho M de tamanho f seria o resultado do algoritmo.