MO417 – Complexidade de Algoritmos

Nathana Facion - RA 191079 - Lista 10

09/06/2017

Questão 1.

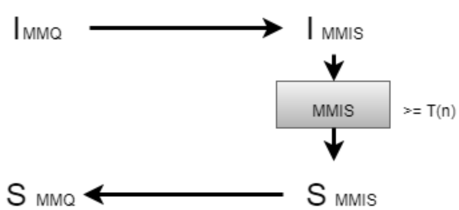
Chamamos de $MMIS$ o problema de multiplicar uma matriz triangular inferior por uma matriz triangular superior.

Entrada: Uma matriz triangular inferior M_i de tamanho n x n, e uma matriz triangular superior M_s de tamanho n x n
Saída: A matriz quadrada M_t de tamanho n x n, como resultado do produto entre M_i e M_s, isto é, $M_t = M_i * M_s$
Complexidade: A complexidade do algoritmo é dada pela função T(n).

Chamamos de MMQ o problema de multiplicar duas matrizes quadradas arbitrárias.

Entrada: Duas matrizes quadradas de tamanho n x n, que chamamos de M_{A1} e M_{A2}.
Saída: Uma matriz quadrada M_{At} de tamanho n x n, como resultado do produto entre M_{A1} e M_{A2}, isto é, $M_{At} = M_{A1} * M_{A2}$
Complexidade: A complexidade do algoritmo é dada pela função G(n).

Queremos provar que MMQ $\prec_{h(t)}$ $MMIS$



O algoritmo para MMQ terá tempo de h(n) + T(n). Considere que G(n) é ótimo para MMQ, então h(n)+T(n) $\geq$ G(n) assintoticamente. Basta então que h(n) $\leq$ o(T(n)).

Considere então que temos duas matrizes A e B de tamanho n x n . Então vamos criar uma nova matriz C e D com tamanho 2n x 2n. Se os elementos que se encontram acima da diagonal principal forem iguais a zero, isto é, se for nulo todo elemento de forma que C_{ij} em que i < j, com isso obtemos uma matriz triangular inferior. Se os elementos que se encontram abaixo da diagonal principal forem iguais a zero, isto é, se for nulo todo elemento de forma que D_{ij} em que i > j , com isso obtemos uma matriz triangular superior. Logo MMIS terá complexidade T(2n) $\in$ O(T(n)). Para criarmos C e D podemos considerar as duas fórmula abaixo:

$$\mathrm{C}_{i,j} = \begin{cases} \mathrm{A}_{i,j}, & \text{para } i > n, j \leq n \\ 0, & \text{senão} \end{cases} \quad (1)$$

$$\mathrm{D}_{i,j} = \begin{cases} \mathrm{B}_{i,j}, & \text{para } i \leq n, j > n \\ 0, & \text{senão} \end{cases} \quad (2)$$

Visualmente o que teremos são as seguintes matrizes:

$$C = \begin{bmatrix} 0 & 0 \\ A & 0 \end{bmatrix} D = \begin{bmatrix} 0 & B \\ 0 & 0 \end{bmatrix}$$

Seja E uma matriz que vai ser o produto dessas matrizes:

$$E = C * D = \begin{bmatrix} 0 & 0 \\ 0 & AB \end{bmatrix}$$

Nós temos então as seguintes transformações:

function $\tau_I(A, B)$

Criação de C:
1) Cria um laço for que vai de 1 a 2n.
2) Dentro cria um outro laço for que vai de 1 a 2n.
3) Segue a fórmula (1) acima para adicionar 0 ou o valor de A em C.

Criação de D:
1) Cria um laço for que vai de 1 a 2n.
2) Dentro cria um outro laço for que vai de 1 a 2n.
3) Segue a fórmula (2) acima para adicionar 0 ou o valor de B em D.

devolve

$$C = \begin{bmatrix} 0 & 0 \\ A & 0 \end{bmatrix} D = \begin{bmatrix} 0 & B \\ 0 & 0 \end{bmatrix}$$

function $\tau_S(E)$

Cria matriz E':

1) Cria um laço for que vai de 1 a n

2) Dentro cria um outro laço for que vai de 1 a n

3) Copia de E[n+1...2n][n+1...2n] para E'.

devolve E'

Vamos mostrar que todos os algoritmos para MMIS tem complexidade Ω(G(n)). Logo, G(n) é uma cota para MMQ.

Como para executar o MMIS com as matrizes C e D, teremos que comparar 2n elementos em C com 2n elementos em D, então teremos, ao final do algoritmo, realizado O(n^2) comparações. Podemos considerar que $n^2 \leq$ T(n).

Como G(n) é uma cota inferior para MMQ, então O(n^2)$+T(2n) \geq \Omega$(G(n)):

$$O(n^2) + O(T(n)) \geq O(n^2) + T(2n) \geq \Omega(G(n))$$

$$O(T(n)) \geq \Omega(G(n)) - O(n^2)$$

Com isso podemos concluir que :

$$O(T(n)) \geq \Omega(G(n))$$

Argumentação: Considere A e B duas matrizes n x n. Considere C e D duas matrizes 2n x 2n, estas são triangulares. Seja E a multiplicação de C e D. Então :

$$C = \begin{bmatrix} 0 & 0 \\ A & 0 \end{bmatrix} D = \begin{bmatrix} 0 & B \\ 0 & 0 \end{bmatrix}$$

$$E = C * D = \begin{bmatrix} 0 & 0 \\ 0 & AB \end{bmatrix}$$

Para obtermos a multiplicação AB, basta $E'_{ij} = E_{i+n,j+n}$. Logo, temos que E'_{ij} é o resultado da multiplicação das matrizes A e B. Como utilizamos MMIS para conseguir o resultado da MMQ, podemos concluir que $MMQ \prec_{h(t)} MMIS$.

Questão 2.

Chamamos de ORD o problema que ordena uma sequência de números.

Entrada: Uma sequência de números, $s_1, s_2, s_3..s_n$.
Saída: Uma sequência de números ordenados, $o_1 \leq o_2 \leq o_3... \leq o_n$.

Chamamos de $AGMC$ o problema de árvore geradora mínima em um grafo completo.

Entrada: G um grafo completo que, dada uma aresta (u,v), o custo dessa aresta é a distância euclidiana entre os vértices u e v.
Saída: A árvore geradora mínima T que tem os custos minimizados.

Queremos provar que ORD $\prec_{h(t)}$ $AGMC$

Considere que ORD tem $\Omega(nlgn)$, então queremos demostrar que AGMC também é $\Omega(nlgn)$

Considere que temos uma vetor A de números $s_1, s_2, s_3..s_n$. Mapearemos esses números em um grafo G. Para isso, criaremos pares de coordenadas da forma (s_k,0) para cada número na posição k em A. Então teremos:

$\tau_I(\{s_1, s_2, s_3..s_n\})$
devolve $\{(s_1, 0), (s_2, 0), (s_3, 0)..(s_n, 0)\}$

A complexidade de τ_I é O(n) , pois cada elemento vai ser criado como uma coordenada.

Então, com o grafo G em mãos, usaremos o algoritmo caixa preta AGMC para encontrar a árvore geradora mínima T de G. Basta então pegarmos o vértice com a menor coordenada x em T e retornarmos os vértices em sequência. Como todos os vértices estavam alinhados, quando encontramos o menor, sabemos que o segundo maior será o próximo, e assim por diante, com isso a ordenação fica evidente.

$\tau_S(\{(s_1, 0), (s_2, 0), (s_3, 0)..(s_n, 0)\})$
1) Encontre v_1 vértice de menor abcissa
2) Retorne, começando por v_1, sequencialmente as abcissas dos vértices.
devolve $\{v_1, v_2, v_3, v_4.., v_n)\}$ // vértices ordenados

A complexidade para percorrer a árvore geradora é O(n). Logo, $h(n) \in$ o(nlogn)

Argumentação: Seja $A = s_1, s_2, s_3..s_n$ um vetor de números a ser ordenados. Criaremos um grafo G a partir desse vetor. Para isso, para cada elemento k de A, mapearemos um vértice da coordenada (S_k,0) em G. Esse mapeamento leva tempo O(n). Com G, executamos o algoritmo AGMC, que retorna uma AGM de G, que chamamos de T. T terá seus vértices ordenados de forma que o vértice na posição i estará mais à esquerda que o vértice na posição $i + 1$, isto é, estará ordenado de forma crescente da coordenada x (já que todo y é 0). Portanto, podemos obter o vetor $A' = \{o_1 \leq o_2 \leq o_3 ... \leq o_n\}$ com valores não descrescentes com os valores de coordenada x dos vértices de T percorrendo da esquerda para a direita. Portanto, podemos concluir que ORD $\prec_{h(t)} AGMC$