Relatório 1º projeto ASA 2021/2022

Grupo: al070

Aluno(s): Luís Freire D'Andrade (94179) e Joana Maria de Brito (96037)

Descrição do Problema e da Solução

O 1º projeto de ASA 2021/2022 é composto por dois problemas. Segundo o enunciado, o problema 1 consiste, dada uma sequência de inteiros ($\vec{x}$), em determinar o tamanho da sua maior subsequência estritamente crescente (LIS), bem como o número de subsequências estritamente crescentes de tamanho máximo. Já o problema 2 resume-se, dadas duas sequências de inteiros ($\overrightarrow{x_1}$ e $\overrightarrow{x_2}$), apenas a calcular o tamanho da maior subsequência comum estritamente crescente entre as duas (LCIS). Ambas as nossas soluções, implementadas em linguagem C++, recorrem a uma estrutura personalizada, composta por um vetor de inteiros e um inteiro que representa o tamanho desse vetor, para a representação das sequências e resolução dos problemas.

A solução do problema 1 é facilmente determinada com recurso a programação dinâmica e a tabulação, com dois vetores/tabelas auxiliares (do tamanho da sequência inicial) que guardam os resultados dos seus subproblemas (referentes ao tamanho e ao número de LIS até, e inclusive, cada índice da sequência inicial), visto que tem caráter recursivo:

$\vec{C}, \vec{N}$: vetores/tabelas auxiliares.

$\vec{C}(i)$: tamanho da maior subsequência estritamente crescente que termina na posição i.

$\vec{C}(i) = max(\vec{C}(j) + 1 \mid 0 \leq j < i \wedge \vec{x}[j] < \vec{x}[i])$

$\vec{N}(i)$: número de subsequências estritamente crescentes de tamanho máximo que terminam na posição i.

$\vec{N}(i) = \sum_j \vec{N}(j) \mid 0 \leq j < i \wedge \vec{C}(i) = \vec{C}(j)$

Dadas duas variáveis ($maxLen$ e $maxNum$, atualizadas aquando a descoberta de uma LIS com um novo tamanho máximo, ou, de uma LIS de tamanho igual à de tamanho máximo) que representam a solução do problema, a ideia é iterar sobre todos os elementos da sequência inicial (ciclo exterior) e, para cada índice, percorrer todas as posições que estão para trás do mesmo (ciclo interior), com vista a preencher os vetores/tabelas e as variáveis da solução:

$$\vec{C}(i) > maxLen: maxLen = \vec{C}(i), maxNum = \vec{N}(i); \quad \vec{C}(i) = maxLen: maxNum = maxNum + \vec{N}(i)$$

O problema 2 recorre também a programação dinâmica e tabulação, com um vetor auxiliar (do tamanho da menor sequência, $\overrightarrow{lcis}$) que representa uma linha de uma matriz C (onde, $C[i][j] = \overrightarrow{lcis}[j]$). Recorre também a duas variáveis ($currLcis$ e $maxLcis$ inicializadas a 0) que simbolizam, respetivamente, o tamanho da atual subsequência comum entre dois vetores e o tamanho da maior, a solução do problema.

$C[i][j]$: tamanho da maior subsequência estritamente crescente comum entre $\overrightarrow{x_1}[\dots i]$ e $\overrightarrow{x_2}[\dots j]$.

Por questões de eficiência, é só passado ao programa os valores da segunda sequência que são comuns com a primeira (e é posteriormente determinada qual a menor e maior sequência para economizar memória no vetor auxiliar). A ideia agora é iterar sobre as duas sequências, de forma encaixada, para conseguir preencher o vetor, considerado apenas o caso em que $\overrightarrow{x_1}[i] == \overrightarrow{x_2}[j]$: $\overrightarrow{lcis}[j] = max(\overrightarrow{lcis}[j], currLcis + 1)$ (se o valor obtido for superior a $maxLcis$, então este é atualizado). Mas quando $\overrightarrow{x_1}[i] > \overrightarrow{x_2}[j]$: $currLcis = max(\overrightarrow{lcis}[j], currLcis + 1)$ e a variável $currLcis$ é atualizada (e cada vez que é percorrido o ciclo interior a variável $currLcis$ é reiniciada a 0).

Relatório 1º projeto ASA 2021/2022

Grupo: al070

Aluno(s): Luís Freire D'Andrade (94179) e Joana Maria de Brito (96037)

Análise Teórica

Nesta análise teórica, considera-se N como o tamanho da primeira sequência de inteiros, e M como o tamanho da segunda (problema 2).

- Leitura dos dados de entrada e construção das sequências: simples leitura do input, com um ciclo a depender linearmente de N (problema 1)/ N + M (problema 2). Logo Θ(N)/ Θ(N + M).
- Resolução do problema 1: criação e inicialização dos vetores auxiliares a 1 [O(N)]; resolução dos seus subproblemas e preenchimento dos vetores [ciclo interior que itera sobre o índice do ciclo exterior até ao início da sequência inicial para preencher uma entrada em cada vetor/tabela O(N), verificações e atualizações O(1)]. Logo, O(N*N) para preencher as N entradas de cada vetor e encontrar a solução.
- Resolução do problema 2: criação e inicialização do vetor auxiliar a 1 O(N); resolução dos seus subproblemas e preenchimento do vetor [dois ciclos encaixados a iterar sobre o tamanho da maior e menor sequência respetivamente O(M*N), verificações e atualizações das variáveis O(1)]. Logo, O(M*N).
- Apresentação dos dados. O(1)

Complexidade global da solução: O(N*N) (problema 1)/ O(M*N) (problema 2)

Avaliação Experimental dos Resultados

Para cada experiência foram gerados 10 grafos de tamanho incremental. De seguida, foi cronometrado o tempo de execução do programa para cada um dos grafos gerados. Como resultado, foram originados os gráficos da Figura 1 e da Figura 2.

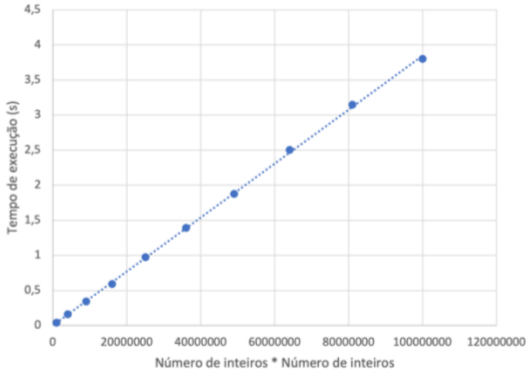

*Figura 1 - Gráfico de Tempo de Execução em função de N*N*

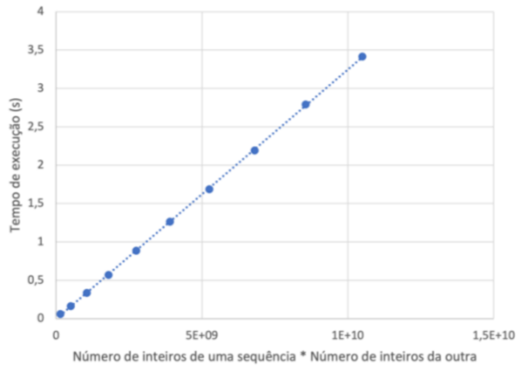

Figura 2 - Gráfico de Tempo de Execução em função de !??!

Sendo que as linhas de tendência linear dos gráficos (com o tempo de execução em função da complexidade do respetivo problema) se revelam bastante próximas de todos os pontos, nos dois casos, pode-se concluir que os gráficos gerados para os diferentes problemas estão concordantes com análise teórica acima descrita, pois é possível observar que o tempo de execução do programa cresce linearmente com a complexidade apontada. Logo, a complexidade global da solução de cada problema verifica-se: O(N*N) (problema 1)/ O(M*N) (problema 2).