Relatório 1º projecto ASA 2022/2023

Grupo: AL049

Aluno(s): Pedro Manuel Afonso Sousa (102664) e Pedro Alexandre Fatia Curto (103091)

Descrição do Problema e da Solução

No relatório, quando surgir O(n) por exemplo, considera-se "n" o lado de um quadrado nxn, uma vez que a maioria dos testes desenvolvidos e relevantes também foram efetuados em quadrados de dimensão nxn. Se surgir "m", refere-se ao número de colunas. Para resolver o problema apresentado neste projeto utilizámos, primeiramente, para guardar os dados de entrada, um vetor de inteiros no qual se guardam todos os c'is diferentes de 0 e 1 (ou seja, os inteiros que representam o índice da coluna pela qual passa o caminho na i-ésima linha). Posteriormente, considerando que o input fornecido é nxm, se o número de linhas (m) for superior ao número de colunas (n), executa-se um procedimento equivalente a transpôr o caminho ou a matriz, uma vez que desse modo o programa é mais eficiente. Depois, recorremos a uma função solução recursiva, que recebe como argumento o vetor de c'is.

Dentro da função recursiva, recorre-se a "memoization", através de uma estrutura (unordered map) em que se guarda o valor de uma determinada "escada"; se a da recursão atual já tiver sido calculado, retorna-se simplesmente esse valor. Caso contrário, calculamos o proximo quadrado a ser analisado (prioritizando o quadrado mais à direita e seguidamente mais acima). De seguida, calculamos o tamanho maximo que um ladrilho pode ter começando no quadrado escolhido e para cada tamanho que o ladrilho possa ter, faz-se uma chamada à função recursiva com a nova "escada" sem o ladrilho escolhido. No final, adiciona-se uma entrada no map com o número dessa "escada".

Análise Teórica

- Leitura dos dados de entrada: loop que depende linearmente de n; logo, Θ(n).
- Apresentação dos dados: O(1).
- Aplicar uma transformação ao vetor de modo a transpôr a matriz: $O(n^2)$
- Calcular quadrado seguinte dentro da função recursiva: O(n)
- Calcular tamanho maximo do ladrilho dentro da função recursiva: O(n)
- Loop dentro do qual se chama a funcao recursivamente: $O(n^2)$

Complexidade global da solução: $O(n^m)$. Este foi o upper bound que, de acordo com os testes feitos e a análise da solução desenvolvida, fazia mais sentido apresentar, apesar de ser bastante exagerado, uma vez que a solução submetida tem otimizações que permitem melhorar o tempo, tal como a utilização de "memoization" que melhorou significativamente o tempo aquando da sua implementação.

Relatório 1º projecto ASA 2022/2023

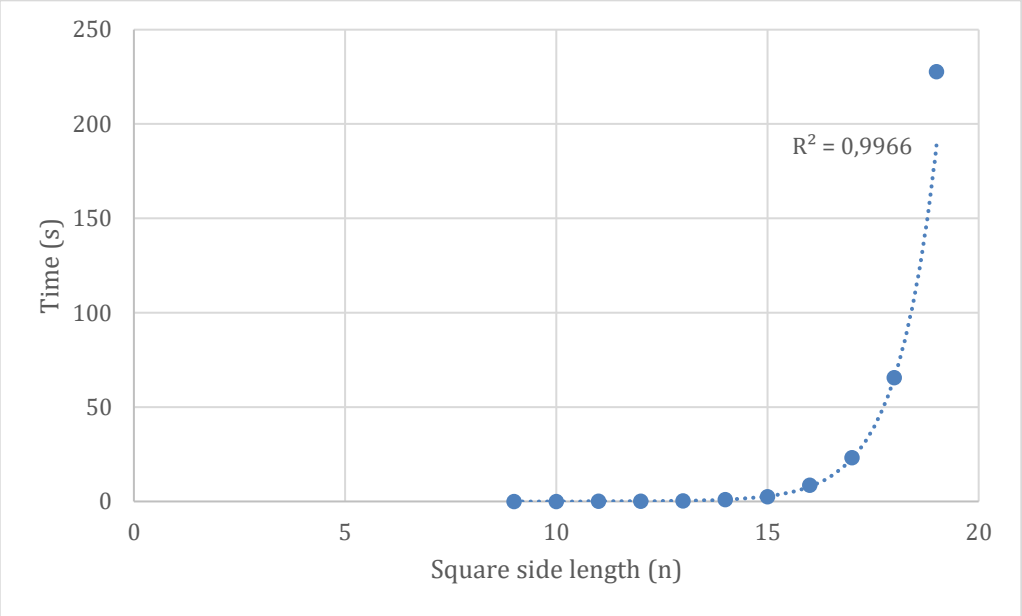
Grupo: AL049

Aluno(s): Pedro Manuel Afonso Sousa (102664) e Pedro Alexandre Fatia Curto (103091)

Avaliação Experimental dos Resultados

Para gerar o gráfico, testámos 11 instâncias de tamanho incremental, de tamanhos nxn desde 9x9 até 19x19. Os tempos para cada instância são os seguintes:

Square size (n)	9	10	11	12	13	14	15	16	17	18	19
Time (s)	0,008	0,014	0,031	0,084	0,302	0,883	2,528	8,672	23,149	65,516	227,74



O gráfico obtido experimentalmente, que relaciona a dimensão n de um dado quadrado nxn utilizado como teste com o tempo que esse mesmo teste demora a ser executado, encontra-se de acordo com a previsão teórica uma vez que apresenta um crescimento exponencial, porém, não descreve uma função n^m, uma vez que, tal como explicado na análise teórica, apesar do *upper bound* ser $O(n^m)$, esse é obtido através da análise puramente teórica do código para o pior caso, ignorando que existem uma série de otimizações que permitem que nunca se aproxime de tal, como a referida "memoization", e outros ajustes e pequenos melhoramentos feitos no código de modo a se conseguir melhores tempos. De acordo com os resultados experimentais, até 19x19, consegue-se descrever a função apresentada como algo próximo de 2^n, sendo que desde esse caso apresenta um crescimento mais acentuado, até se aproximar eventualmente da previsão teórica.