Relatório 1º projecto ASA 2022/2023

Grupo: AL039/TP11

Aluno(s): Gonçalo Rua (102604) e João Gouveia (102611)

Descrição do Problema e da Solução

A solução que encontrámos passa por encontrar o vértice mais à direita de um dado tabuleiro. Depois, encontramos o maior quadrado que seja possível inserir nesse canto e, assim, sabemos que os quadrados que podemos inserir nesse sítio são aqueles de medida entre a maior e 1. Seguidamente geramos tabuleiros novos, cada um com um dos quadrados possíveis inseridos no seu canto mais à direita. O número de maneiras de ladrilhar o tabuleiro pedido vai ser a soma do número de maneiras de ladrilhar os novos tabuleiros criados. Obviamente, vão haver vários tabuleiros repetidos. Para evitar calcular várias vezes as combinações destes tabuleiros, guardamos todos os tabuleiros cujo valor das possibilidades de ladrilhar já foi calculado numa hash table, aumentando assim a eficiência do nosso algoritmo. Para colisões foi usado rehashing.

Análise Teórica

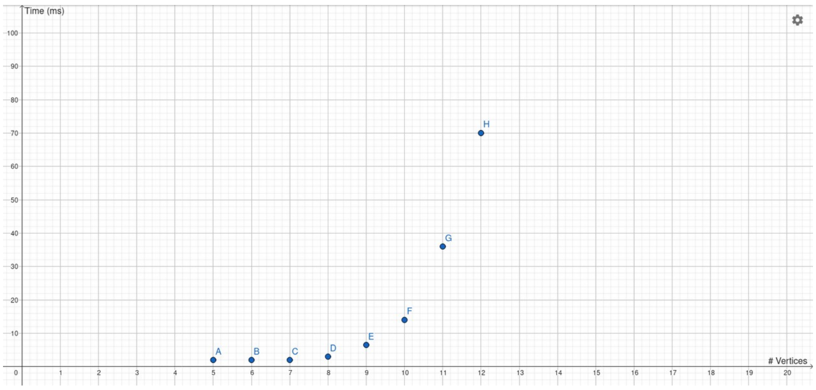
A complexidade total do algoritmo é $O(X^2)$, sendo X o número de colunas do tabuleiro. Isto porque a leitura da entrada é $O(Y)$, mas o algoritmo roda o tabuleiro (troca o Y com o X) sempre que $Y >> X$. Abaixo encontra-se uma análise mais promenorizada de cada função, com o seu respetivo pseudocódigo.

- Leitura de dados de entrada (read_input): Simples leitura do input, com um ciclo que depende da altura (Y) do tabuleiro, $\Theta(Y)$.
- Função inicial do problema (compute_board): Faz uso da auxiliar null_board, $O(Y)$, e da função auxiliar spin_board, $O(Y)$, nos casos em que o tabuleiro é um retângulo com $Y > X$. Chama a função get_combinations, $O(X^2)$, para obter o resultado do problema. Como rodamos o tabuleiro quando $Y > X$, esta função é $O(X^2)$.
- Função auxiliar para determinar se o tabuleiro tem todos os cantos com valor zero (null_board): Função com um ciclo que percorre as linhas e termina se encontrar um canto com valor diferente de 0, logo $O(Y)$. No entanto, na maior parte das vezes o tempo de execução desta função será bastante menor.
- Rodar o tabuleiro (spin_board): O primeiro loop tem no máximo X iterações, pelo que é $O(X)$. O segundo loop terá X ou Y iterações, dependendo do tabuleiro a tratar. No entanto, sabemos que esta função apenas é chamada quando $Y > X$, logo, o segundo loop é $O(Y)$. Quanto ao último loop, tal como o primeiro, tem no máximo X iterações e, por conseguinte, é $O(X)$. Atendendo ao facto desta função só ser chamada quando $Y > X$, podemos concluir que é $O(Y)$.
- Função principal que obtém o resultado (get_combinations): finished_board é $O(Y)$, lookup_board e insert_board são $O(1)$ em média e place_rightmost_tile é $O(X^2)$. children tem no máximo X elementos, uma vez que, devido á função spin_board, $X > Y$ (caso isto não se verifique é porque a diferença é mínima e podemos afirmar, aproximadamente, $X = Y$). Assim, temos $T(X) = XT(X^2 - b) + O(X^2)$, pois, mais uma vez, $X > Y$. Logo, assumindo um b insignificante, temos que esta função é $O(X^3)$ no caso em que não se repetem casos. Considerando o uso da hashtable para casos repetidos, verificamos que, em média, 1/3 dos casos já lá se encontram, concluindo que esta função é $O(X^2)$.
- Verifica se chegámos ao fim do cálculo do tabuleito passado como argumento

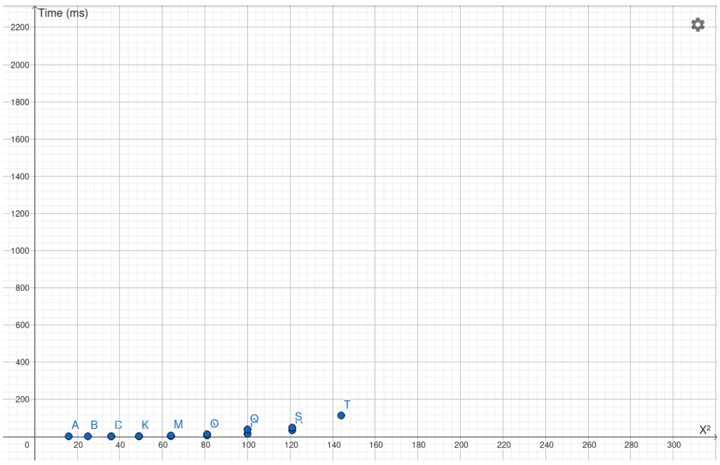
(pois o maior quadrado já só pode ter tamanho 1 ou 0) (finished_board): Função com um ciclo que percorre as linhas e termina se encontrar um canto com valor superior a 1, logo O(Y).
- Verifica se dois tabuleiros são iguais (equal_boards): No pior caso são iguais e a função terá de percorrer todos os corners, que corresponde ao número de linhas do tabuleiro, logo O(Y).
- place_rightmost_tile: O primeiro loop é O(Y) e o segundo depende do tamanho do ladrilho máximo, pelo que é O(X) ou O(Y). O terceiro loop também depende do tamanho do ladrilho máximo, que é O(X^2) ou O(Y^2) pois é o (somatório de i=0 até tile_size – 1) do (somatório de j=0 até i-1). new_corners terá no máximo X ou Y elementos. Sabemos que regra geral, devido á função spin_board, X > Y (caso isto não se verifique é porque a diferença é mínima e podemos afirmar, aproximadamente, X = Y), com vista a isto, conclui-se que esta função é O(X^2).

Avaliação Experimental dos Resultados

Para a avaliação experimental, foram gerados testes progressivamente maiores (referidos no fim do documento). A temporização da duração da resolução de cada teste foi feita com recurso ao comando "time". Logicamente, o gráfico não é linear quando o eixo X varia de acordo com o número de vértices.



Trocando a variação deste eixo para uma coerente com o que foi proposto teoricamente (X^2), verificamos rapidamente que se trata de um gráfico linear, o que corrobra a nossa análise teórica.



Abaixo estão os testes utilizados (todos os tabuleiros completamente preenchidos, isto é, os cantos têm todos o valor da largura do tabuleiro):
10x10, 10x9, 9x9, 9x8, 8x8, 8x7, 7x7, 7x6, 6x6, 6x5, 5x5, 5x4