Report for Programming Problem 2 - ARChitecture

Team:

Student ID: 2018297281 Name: Diogo Valente Martins

Student ID: 2018287956 Name: Miguel Alves Pimenta

1. Algorithm description

O algoritmo consiste em três ciclos um para percorrer as colunas e outro para percorrer as linhas este ciclo de percorrer as linhas é utilizado duas vezes dentro de uma iteração do ciclo de percorrer as colunas, para calcular os valores da coluna na subida e posteriormente para calcular os valores da coluna na descida. Na matriz que é inicializada apenas com 0´s quando colocamos um lego representamos sempre a posição do lego na sua altura, por exemplo se tivermos um lego que o h é 3, então na primeira coluna vai se colocar um 1 na linha 3 e assim sucessivamente, e se nessa posição for possível ter mais de uma combinação ou seja mais de um lego que origine um arco diferente então sumamos um ao valor existente já nessa posição. Então inicialmente começamos com o caso base, ou seja na primeira coluna colocamos o número um na posição da altura do lego, pois só há uma possibilidade e a partir daí começamos a calcular todas as próximas colunas com base na anterior, isto é somamos os números que estão na linha imediatamente antes na mesma coluna e na coluna anterior, temos ainda de subtrair o valor de h linhas abaixo da coluna anterior e assim fica representada a subida, na descida a forma de calcular os valores são exatamente iguais mas os índices são os inversos e para além disso temos de ter em atenção também os valores da subida o que não acontecia a calcular as subidas com os valores da descida. Depois a solução era nos dada pelos valores que estavam em todas as colunas exceto a primeira nas linhas de índice h. Basicamente baseamo-nos nos valores anteriores para obter os valores das colunas que estávamos a calcular. Isto foi necessário, pois inicialmente tínhamos 3 ciclos porque percorríamos as colunas duas vezes para calcular os valores ao fazer isto só com dois e baseando-nos nos valores obtidos anteriormente resolvemos alguns problemas de tempo para além disto tivemos também de limitar até onde iam os ciclos das linhas, porque para h pequenos estávamos a percorrer demasiadas linhas com 0 que nada alteravam na solução então temos uma formula que nos dá a altura máxima das linhas que vale a pena calcular. Para além de problemas de tempo tivemos de memória que irei explicar no ponto dois como foi corrigido pois tem haver com a estrutura utilizada.

2. Data structures

Inicialmente utilizamos um vector de três dimensões em que o número de linhas era H, colunas o n, e a terceira dimensão tinha tamanho 2 para representar a sala a subir e outra para representar a sala a descer, com isto deparamo-nos com um problema de memória excedida, para resolver isto em vez de utilizarmos n colunas passamos a utilizar apenas 3 e sempre que executamos os cálculos

colocamos a coluna que não está a ser utilizada a 0 ao mesmo tempo que efetuamos os cálculos nas outras.

3. Correctness

Eu considero a minha abordagem correta, pois consegui reduzir o número de iterações do segundo ciclo calculando uma altura máxima até onde podia iterar e valia a pena calcular, e para evitar fazer um terceiro ciclo calculo os valores que preciso com base em valores das linhas anteriores já calculados. Para além disso reduzi bastante a memória da minha abordagem usando apenas seis colunas três para descer e três para subir, ambas as colunas com altura H.

4. Algorithm Analysis

(Discuss the overall memory and time complexity of your approach.)

A complexidade do temporal da minha abordagem é O(n*maxH) sendo n o número de colunas e maxH o número de iterações que são realizadas nas linhas que é calculada, isto é neste ciclo só iteramos o que seja necessário e relevante para os cálculos e nunca as linhas todas existentes na matriz.

5. References

Para a resolução deste problema foram as funções dos módulos fornecidas pelos professores da cadeira e também os slides fornecidos pelos mesmos nos materiais de apoio.