Universidade de Évora Estruturas de Dados e Algoritmos II Ano Letivo 2021/2022

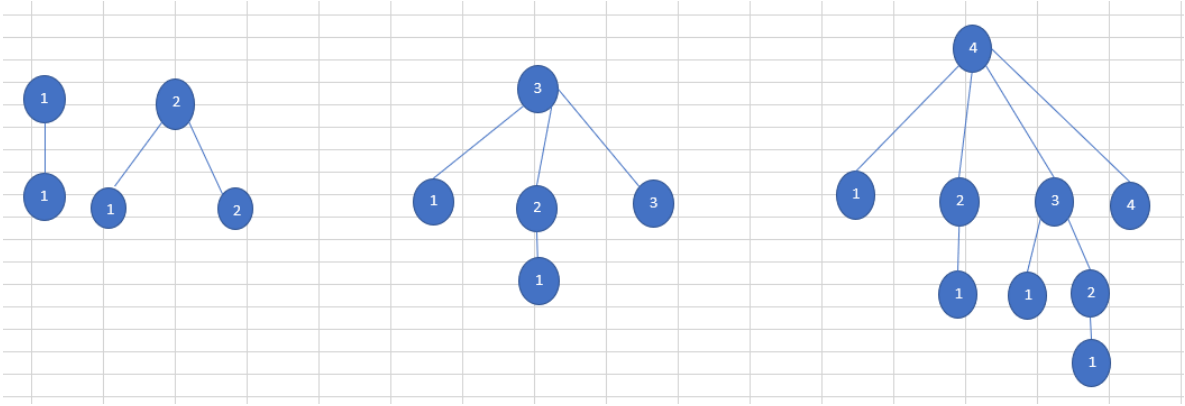
Miguel Correia 37884
Miguel Calado 48398
Mooshak:g104

1 Introdução

Este relatório é referente ao problema "Mosaics". O objetivo do trabalho é construir um mosaico em lego com tamanho definido por input. Cada peça tem tamanho de comprimento variável e altura igual a 1.Dados os tamanhos e cores de cada peça, temos de calcular o número total de combinações possíveis usando apenas peças de tamanho previamente definido.

2 Problema

Primeiramente começámos por definir qual seria a melhor maneira de abordar o problema de forma geral, e chegámos à conclusão que a maneira mais eficiente seria calcular as possibilidades de cada linha e multiplica-las entre si. Dentro de cada linha começamos por contar quantas peças iguais há de seguida, e calculamos as possibilidades para essa peça, depois, repetimos o processo para todas as peças da linha e multiplicamos os resultados entre si, e assim obtemos as possibilidades da linha.



3 Algoritmo

De modo a implementar o programa com a abordagem descrita no ponto anterior começamos por percorrer cada linha do input individualmente, em seguida, se a peça na posição 1 for igual à peça na posição 2 adicionamos 1 ao contador (que já era previamente igual a 1). Repetimos este processo até que a peça seguinte seja diferente. Nesta altura temos o número de peças iguais consecutivas para uma parte da linha. O próximo passo é agora calcular as possibilidades para esse valor(tamanho_peca), que vai ocorrer do seguinte modo: Tal como está ilustrado na figura 1. começamos por subtrair a "tamanho_peca" a menor peça possível, até chegarmos à maior peça possível, menor do que "tamanho_peca", por exemplo, se "tamanho_peca" for 7, a menor peça possível é 1 e a maior peça possível é 6. Tendo em conta que o resultado 7-6=1, concluímos que o número de possibilidades de fazer de fazer aquele mosaico quando a primeira peça que usamos é a peça de tamanho 6 são as possibilidades de 1, que é igual a 1(para fazer um mosaico de 1 só podemos usar a peça de tamanho 1). Repetimos este processo para todos os tamanhos de peça possíveis menores que o "tamanho_peca".
De modo a termos sempre disponíveis os valores que precisamos começamos sempre por calcular as possibilidades desde 1 até ao valor pretendido, ou seja, se "tamanho_peca" igual a 7, iremos calcular as possibilidades de 1 até 7(a não ser que estas já estejam guardadas em memória, nesse caso não são calculadas novamente), de modo a que, por exemplo, no momento em que subtraímos a peça de tamanho 1 tenhamos disponível o valor das possibilidades da peça de 6.
Por ultimo, "possibilidades[tamanho_peca]" vai ser igual à soma de todos os valores de "possibilidades[tamanho_peca – pecas]", ou seja, no caso de "tamanho_peca" igual a 7, as possibilidades de 7 vai ser a soma de "possibilidades[tamanho_peca -1]", "possibilidades[tamanho_peca -2]" e assim sucessivamente até "possibilidades[tamanho_peca -6]". Destas somas resulta o valor das possibilidades[tamanho_peca], valor esse que é guardado num array, para caso seja necessário outra vez no decorrer do programa não ter de ser calculado novamente.

1

4 Recursiva

Função calcula

```
Let possibilidades[tamanho_peca] be a new array
possibilidades[1]=1
for i=1 to tamanho_peca do
    for k=1 to pecas do
        if i>k then
           possibilidades[i] = possibilidades[i] + calcula(possibilidades[i-k])
        else if i==k then
           possibilidades[i] = possibilidades[i] + 1
```

6 Complexidade Temporal

Quando o programa inicia são iniciados 2 ciclos for, um deles corre "lines" vezes, e o outro corre "cols-1" vezes, de modo a percorrer todas as linhas e colunas do input, logo, inicialmente a complexidade é O(lines) * O(cols) = O(lines*cols). Dentro destes dois ciclos estão mais dois ciclos, da função "calcula". O primeiro ciclo corre "tamanho_peca" vezes e o seguinte corre "pecas" vezes, que no pior caso serão 9 vezes, logo a complexidade da função calcula será O(tamanho_peca) * O(peca) = O(tamanho_peca *peca). Tendo em conta que todos os ciclos estão contidos uns nos outros a complexidade será O(lines*cols)*O(tamanho_peca*peca) = O(lines*cols*tamaho_peca*peca). Ou seja, no pior caso a complexidade será O(n^3), em que lines = cols = tamanho_peca = n;

7 Complexidade Espacial

pecas - 9+4bytes = 36bytes
possibilidades - cols+1 * 8byes, pior cenário = 8008bytes
contador - 4bytes
resultado – 8bytes
i,i,k,peca – 4+4bytes=16bytes
No pior cenário temos 8072bytes ocupados