**Mestrado Integrado em Engenharia Informática**

**Análise e Desenho de Algoritmos**

Drina’s Drone

Programação dinâmica

Regente: Doutora Margarida Mamede, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa

Docente: André Rijo, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa

**Turno Prático 1**

**Grupo 51749\_52597**

Andreia Mendes nº51749

Teresa Monteiro nº52597



**Março de 2020**

Indice

Resolução do Problema

Legenda:

i = número de ofertas

L(i) = Lucro da oferta i

P(i) = Lucro máximo possível

possibilities(i) = número de ofertas contiguas até i (já aceites e aceitam i)

No caso base (i = 1) verificamos o caso em que só temos uma oferta que retorna o lucro da mesma.

No caso geral (i > 1) verificamos os casos em que temos mais que uma oferta. Retornamos o máximo entre lucro de i com o lucro máximo obtido anteriormente e o P(i-1), porque queremos que na última posição fique sempre o valor máximo de lucro, o que implica que, mesmo que o P(i) não seja o máximo possível vamos "ignorar" essa oferta e continuamos a partir do máximo já encontrado.

Complexidade Temporal

public void addOffer(int start, int duration, int cost); Θ(1)

public int maxProfit(); Θ(nOffers)

private void ordList(); Θ (n log(nOffers)) ?

private int possibilities(int i); Θ(n) ?

nOffers = é o número de ofertas que a Drina vai ter para um determinado caso

Complexidade Espacial

public void addOffer(int start, int duration, int cost); O(nOffers)

public int maxProfit(); Θ(nOffers + 1) , o +1 corresponde ao zero artificial

private void ordList(); ?

private int possibilities(int i); ?

Conclusão

Durante a execução do problema surgiram nos algumas alternativas para a resolução do problema. Pensamos em outras estruturas de dados, como também outras formas de ordenação dos tempos das ofertas.

Como aspeto positivo do nosso trabalho podemos apontar as estruturas de dados escolhidas, nomeadamente a ArrayList para todas as ofertas, que permitiu uma melhor eficiência na ordenação. No caso da estrutura de dados vetor (vecMaxProfit), uma vez que só trabalhamos com inteiros e queremos apenas inserir e consultar, numa certa posição, achamos que foi a estrutura de dados mais adequada para a resolução do problema.

Aspeto que poderíamos melhorar no desenvolvimento de um algoritmo de ordenação com menor complexidade.

Anexo









