



Trabalho de grupo 2 - DA

Agência de Viagens

Grupo 41

Trabalho realizado por:

David Ferreira, 202006302

Dinis Sousa, 202006303

Francisca Silva, 202005140

Regentes:

Ana Paula Tomás

Rosaldo Rossetti

Práticas:

Vanessa Silva



Índice

- 3 - Descrição do Problema
- 4 - Cenário 1.1: Maximizar a dimensão do grupo
- 5 - Cenário 1.2: Maximizar a dimensão do grupo e minimizar os transbordos
- 6 - Cenário 2.1: Encaminhamento para um grupo
- 7 - Cenário 2.2: Corrigir um encaminhamento
- 8 - Cenário 2.3: Dimensão máxima do grupo
- 9 - Cenário 2.4: Quando é que o grupo se reuniria novamente
- 9 - Cenário 2.5: Os locais do tempo máximo de espera
- 10 - Output final de cada cenário
- 11 - Algoritmo de destaque
- 12 - Principais dificuldades, esforço de cada um



Descrição do Problema

Com este problema pretendemos ajudar uma empresa de agência de viagens a ter um sistema para ajudar na gestão dos pedidos de transportes de grupos de pessoas de local de origem para um local de destino.

De cada Viagem sabe-se o local de origem, local de destino, capacidade do veículo e a duração da viagem.



Cenário 1.1: Maximizar a dimensão do grupo

Problema : Maximizar a dimensão do grupo, sem que o grupo se separe,e indicar um qualquer encaminhamento.

Solução: Neste cenário utilizamos o algoritmo de dijkstra para descobrir o caminho de capacidade máxima.

$$Sol = \gamma : \forall \gamma^*, cap(\gamma^* \leq \gamma)$$

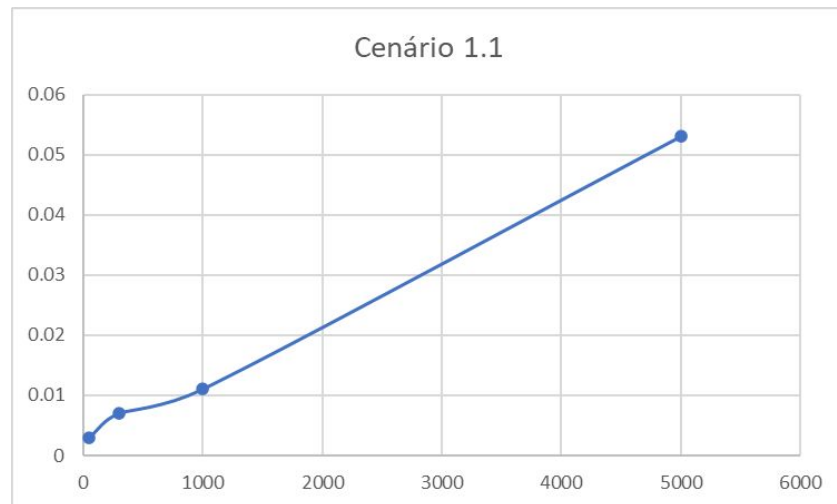
Cenário 1.1: Maximizar a dimensão do grupo

Complexidade Temporal: $O((|V| + |E|) \log_2 |V|)$ (tal como o algoritmo de Dijkstra)

Complexidade Espacial: $O(V)$, (Heap com V entradas).

Avaliação empírica:

```
File 1 - 0.003s (5)
File 3- 0.007s (17)
File 5- 0.011s (16)
File 9- 0.053s (18)
```





Cenário 1.2: Maximizar a dimensão do grupo e minimizar os transbordos

Problema: Maximizar a dimensão do grupo e minimizar o número de transbordos, sem que o grupo se separe, e sem privilegiar um dos critérios relativamente ao outro.

Solução: Usando uma heap de máximos em que a key é um par de distância e nó, e o valor pela qual são ordenados é a capacidade, aplicar uma versão do algoritmo usado em 1.1, tendo em conta as restrições.

Pré-condições:

$$flow(bfs) \leq flow(\gamma^*) \leq flow(CCM) \quad (2)$$

$$switches(bfs) \leq switches(\gamma^*) \leq switches(CCM) \quad (3)$$

CrITÉRIOS de seleção (tem de respeitar ambas as condições):

$$optimal(\gamma) \Rightarrow \forall \gamma^* : switches(\gamma^*) < switches(\gamma), flow(\gamma) > flow(\gamma^*) \quad (4)$$

$$optimal(\gamma) \Rightarrow \forall \gamma^* : switches(\gamma^*) = switches(\gamma), flow(\gamma) \geq flow(\gamma^*) \quad (5)$$

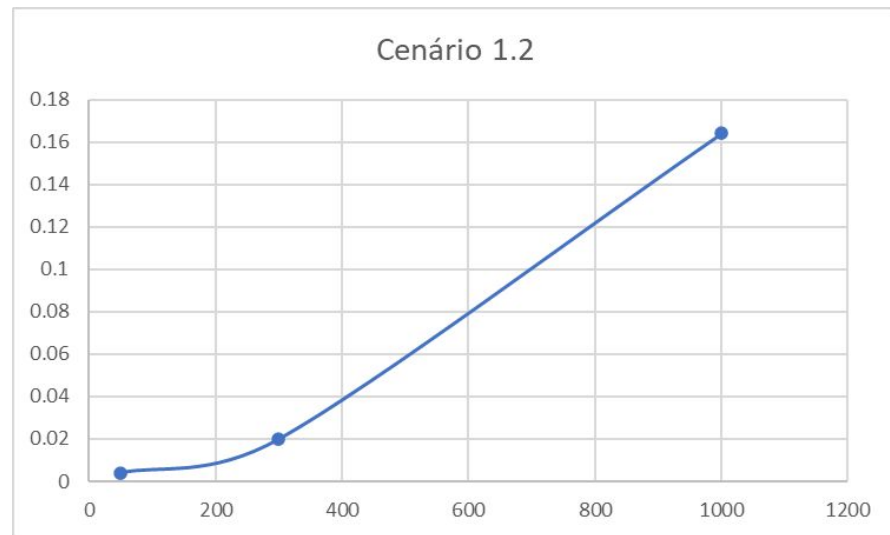
Cenário 1.2: Maximizar a dimensão do grupo e minimizar os transbordos

Complexidade Temporal: $O(?)$

Complexidade Espacial: $O(?)$

Avaliação empírica:

```
File 1 - 0.004s (4.1, 5.2, 6.5)
File 3- 0.020s (4.7, 7.15, 8.16, 17.17)
File 5- 0.164s (5.12, 6.13, 8.15, 10.16)
File 9- NaN
```





Cenário 2.1: Encaminhamento para um grupo

Problema: Determina um encaminhamento para um grupo.

Solução: Neste cenário utilizamos o algoritmo Edmonds-Karp para colocar o máximo de pessoas possível numa rede de fluxo até não existir nenhum caminho com capacidade residual.

2 Cenário 2

Válido para qualquer rede de fluxo:

$$f(u, v) = -f(v, u), u \wedge v \in V \quad (6)$$

$$f(u, v) \leq c(u, v), u \wedge v \in V \quad (7)$$

$$\sum (v \in V) f(u, v) = 0, u \in V \setminus \{s, t\} \quad (8)$$

$$|f| = \sum (v \in V) f(s, v) = \sum (v \in V) f(v, t) \quad (9)$$

$$c_f(u, v) = c(u, v) - f(u, v) \quad (10)$$

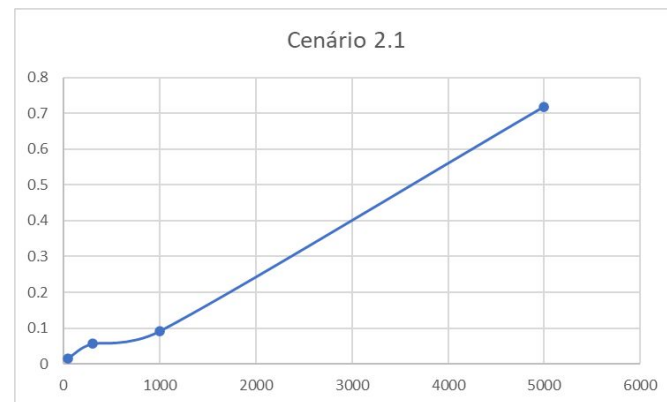
Cenário 2.1: Encaminhamento para um grupo

Complexidade Temporal: $O(m^2n + n) = O(m^2n)$, onde m é o número de arestas e n é o número de nós(algoritmo Edmonds-Karp), e o n somado é do bfs usado para obter os vários caminhos possíveis.

Complexidade Espacial: $O(n + nm) = O(nm)$, onde n é o número de nós(complexidade do bfs), e m é o número de arestas(espaco ocupado pelo grafo auxiliar para retirar os caminhos).

Avaliação empírica:

```
File 1 - 0.016s  
File 3- 0.056s  
File 5- 0.091s  
File 9- 0.719s
```





Cenário 2.2: Corrigir um encaminhamento

Problema: Corrigir um encaminhamento, se necessário aumentar a dimensão do grupo

Solução: No seguimento do cenário anterior recebemos uma mensagem que pergunta se pretendemos adicionar mais pessoas ao grupo.

2.1 Cenário 2.1/2.2

$$|f| = N(\text{Passengers}), N(\text{Passengers}) \leq \max(|f|) \quad (11)$$



Cenário 2.2: Corrigir um encaminhamento

Complexidade Temporal: $O(m^2n) = O(m^2n)$, onde m é o número de arestas e n é o número de nós(algoritmo Edmonds-Karp), e o n somado é do bfs usado para obter os vários caminhos possíveis.

Complexidade Espacial: $O(n + nm) = O(nm)$ onde n é o número de nós(complexidade do bfs), e m é o número de arestas(espaco ocupado pelo grafo auxiliar para retirar os caminhos).

Avaliação empírica:



Cenário 2.3: Dimensão máxima do grupo

Problema: Determinar a dimensão máxima que um grupo pode ter

Solução: Neste cenário utilizamos o algoritmo Edmonds-Karp para colocar o máximo de pessoas possível numa rede de fluxo até não existir nenhum caminho com capacidade residual.

2.2 Cenário 2.3

$$\max(|f|) \iff \forall \gamma(s, t) \in G_f, c(\gamma) = 0 \quad (12)$$

$$|f| = \max(|f|) \quad (13)$$

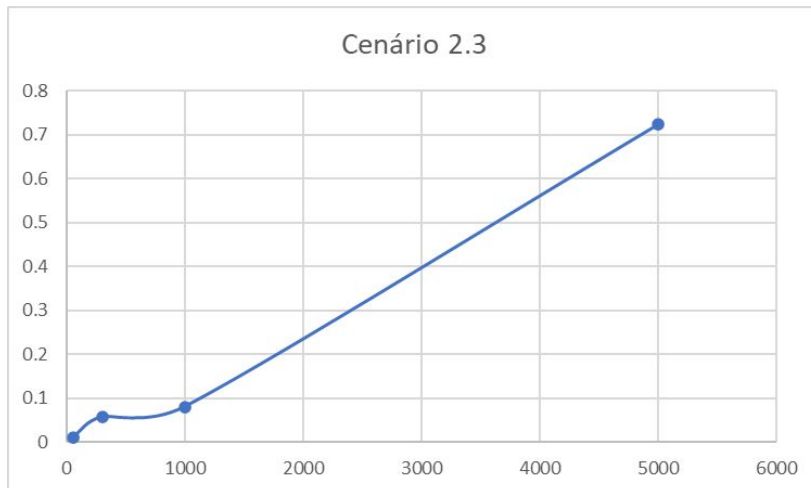
Cenário 2.3: Dimensão máxima do grupo


Complexidade Temporal: $O(m^2n)$, onde m é o número de arestas e n é o número de nós.

Complexidade Espacial: $O(n + nm) = O(nm)$, onde n é o número de nós (complexidade do bfs), e m é o número de arestas (espaço ocupado pelo grafo auxiliar para retirar os caminhos).

Avaliação empírica:

```
File 1 - 0.011s (10)
File 3- 0.058s (104)
File 5- 0.081s (116)
File 9- 0.723 (91)
```





Cenário 2.4: Quando é que o grupo se reuniria novamente

Problema : Partindo de um caminho que constitui um grafo indica quando é que um grupo se reuniria novamente no destino, no mínimo.

Solução: Neste cenário utilizamos algoritmo Edmonds-Karp para colocar o máximo de pessoas possível, posteriormente utilizamos uma função chamada “earliestStart” que vai retornar o tempo em que o último grupo chega ao destino, no mínimo.

2.3 Cenário 2.4

$$ES_i = \max_{(i,j) \in A} \{EF_j\} \quad (14)$$

$$EF_i = ES_i + dur_i \quad (15)$$

$$ES_s = 0 \quad (16)$$

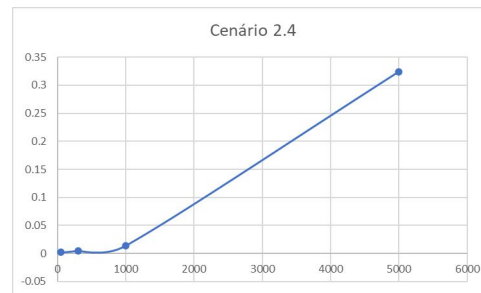
Cenário 2.4: Quando é que o grupo se reuniria novamente

Complexidade Temporal: $O(m^2n + n + n + n) = O(m^2n)$, onde m é o número de arestas e n é o número de nós(algoritmo Edmonds-Karp), e os n 's somados são do bfs usado para obter os vários caminhos possíveis, do algoritmo “earliestStart”, e de uma função usada para diminuir o grafo se o nó de início for diferente de 1.

Complexidade Espacial: $O(n + nm + n + n) = O(nm)$, onde n é o número de nós(complexidade do bfs, do “earliestStart”, e da função para inícios diferentes), e m é o número de arestas(espaco ocupado pelo grafo auxiliar para retirar os caminhos).

Avaliação empírica:

```
File 1 - 0.002s (135)
File 3- 0.005s (180)
File 5- 0.014s (133)
File 9- 0.324 (189)
```





Cenário 2.5: Os locais do tempo máximo de espera

Problema : Admitindo que os elementos do grupo saem todos do mesmo local e partem à mesma hora(o mais cedo possível), indica o tempo máximo de espera e os locais que haveria elementos que esperam esse tempo.

Solução: Calcular os tempos de espera em cada nó do percurso.

2.4 Cenário 2.5

$$WaitTime_{ij} = ES_j - ES_i - dur_{ij} \quad (17)$$

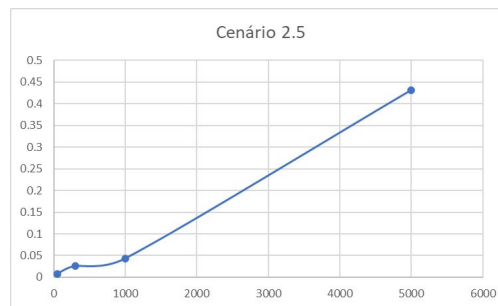
Cenário 2.5: Os locais do tempo máximo de espera

Complexidade Temporal: $O(m^2n + n + n + n + n) = O(m^2n)$, onde m é o número de arestas e n é o número de nós(algoritmo Edmonds-Karp), e os n 's somados são do bfs usado para obter os vários caminhos possíveis, do algoritmo “earliestStart”, de uma função usada para diminuir o grafo se o nó de início for diferente de 1, e de outra para mostrar os tempos de espera.

Complexidade Espacial: $O(n + nm + n + n) = O(nm)$, onde n é o número de nós(complexidade do bfs, do “earliestStart”, e da função para inícios diferentes), e m é o número de arestas(espaco ocupado pelo grafo auxiliar para retirar os caminhos).

Avaliação empírica:

```
File 1 - 0.008s (99 at 50)  
File 3- 0.026s (147 at 300)  
File 5- 0.043s (109 at 1000)  
File 9- 0.432s (77 at 731)
```





Output no final cenário 1

Cenário 1.1:

```
Choose one of the following scenarios:
1.1 / 1.2 / 2.1 / 2.3 / 2.4 / 2.5
(Scenario 2.2 included in 2.1)
1.1
Enter the number of the file to be tested:
1
Capacity: 5
1 6 10 36 17 39 50
New cenário? (y/n)
n

Process finished with exit code 0
```

Cenário 1.2:

```
Choose one of the following scenarios:
1.1 / 1.2 / 2.1 / 2.3 / 2.4 / 2.5
(Scenario 2.2 included in 2.1)
1.2
Optimal solutions:
Switches: 4 Capacity: 1
Switches: 5 Capacity: 2
Switches: 6 Capacity: 5
```

Output no final cenário 2

Cenário 2.1|2.2:

```
Choose one of the following scenarios:
1.1 / 1.2 / 2.1 / 2.3 / 2.4 / 2.5
(Scenario 2.2 included in 2.1)
2.1
Enter the number of the file to be tested:
3
How many people will travel?
5
For flow of 5 people:
1 91 178 285 300
Add more people? (y/n)
y
How many people to add?
10
Add these paths to the previous answer:
For flow of 8 people:
1 91 178 285 300
For flow of 8 people:
1 201 19 46 242 286 300
Add more people? (y/n)
n
New cenário? (y/n)
n
Process finished with exit code 0
```

Cenário 2.3:

```
Choose one of the following scenarios:
1.1 / 1.2 / 2.1 / 2.3 / 2.4 / 2.5
(Scenario 2.2 included in 2.1)
2.3
Enter the number of the file to be tested:
4
For flow of 9 people:
1 150 40 93 300
For flow of 2 people:
1 292 36 66 300
For flow of 10 people:
1 64 100 277 31 300
For flow of 1 person:
1 129 145 140 28 300
For flow of 1 person:
1 129 145 214 31 300
For flow of 6 people:
1 129 145 256 28 300
For flow of 1 person:
1 150 83 8 162 300
For flow of 1 person:
1 198 109 255 92 300
For flow of 7 people:
1 59 141 178 41 162 300
For flow of 2 people:
1 59 141 263 8 162 300
For flow of 3 people:
1 59 251 83 8 162 300
For flow of 2 people:
1 59 251 168 135 234 300
For flow of 1 person:
1 59 297 83 8 162 300
For flow of 4 people:
1 64 288 112 79 93 300
For flow of 3 people:
1 150 146 112 79 93 300
For flow of 2 people:
1 177 58 115 128 93 300
For flow of 2 people:
1 177 74 50 2 162 300
For flow of 2 people:
1 177 146 112 79 285 300
For flow of 3 people:
1 239 216 44 232 221 300
For flow of 4 people:
1 239 251 168 180 92 300
For flow of 2 people:
1 239 251 184 299 28 300
For flow of 8 people:
1 239 287 125 208 221 300
For flow of 1 person:
1 239 287 233 296 255 92 300
Maximum capacity of the group: 77
```



Output no final cenário 2

Cenário 2.4:

```
Choose one of the following scenarios:
1.1 / 1.2 / 2.1 / 2.3 / 2.4 / 2.5
(Scenario 2.2 included in 2.1)
2.4
Enter the number of the file to be tested:
6
The group would reunite at 134

New cenário? (y/n)
n
```

Output no final cenário 2

Cenário 2.5:

Choose one of the following scenarios:

1.1 / 1.2 / 2.1 / 2.3 / 2.4 / 2.5

(Scenario 2.2 included in 2.1)

2.5

Enter the number of the file to be tested:

9

After going from node [n] to node [w] there is a waiting time of [t] at [w] to continue.

[n] -> [w] : [t]

74->2777:37

103->256:22

147->4352:71

241->228:4

279->3812:4

366->1461:35

380->1098:15

398->3371:46

440->2777:26

476->240:35

527->5000:25

616->3526:8

770->4079:37

800->230:8

878->3364:24

881->840:47

977->2769:57

982->1852:60

1034->1098:43

1063->4444:14

1100->731:77

1114->4813:29

1171->1063:45

1212->2769:45

1229->2049:34

1282->1674:1

1284->4257:48

1322->3637:10

1359->2199:25

1408->2268:22

1484->3934:9

1488->3526:24

1569->1674:8

1629->2777:2

1837->440:5

1952->2080:58

1952->3650:72

2010->294:32

2013->1098:27

2049->5000:40

2067->4352:70

2136->3315:15

2182->1488:2

2215->4231:11

2237->658:29

2238->523:17

2296->2012:20

2365->5000:51

2493->3304:16

2603->398:5

2620->1270:24

2631->398:41

2675->1737:15

2706->1573:7

2739->398:45

2755->1357:37

2777->3488:22

2786->3637:14

2807->1488:3

2827->2630:1

2864->3526:36

2894->3488:30

2896->4304:24

2927->2777:25

2934->4555:28

3137->279:41

3166->669:60

3180->4202:15

3203->1540:56

3216->658:32

3364->5000:9

3437->618:37

3457->1071:24

3526->5000:23

3577->3364:57

3630->418:41

3637->745:27

3713->3364:56

3758->745:13

3758->4730:8

3812->5000:41

3904->2630:61

4003->1015:16

4009->1852:43

4038->2209:6

4065->5000:12

4087->4079:68

4117->1674:11

4136->1852:28

4159->1270:29

4164->4576:13

4225->808:49

4231->279:31

4257->3775:7

4265->4903:45

4407->398:63

4590->527:38

4641->3061:2

4680->3154:47

4752->4296:10

4778->4590:3

4842->3488:51

4893->503:23

4958->3136:10

4963->578:40

The maximum waiting time is 77 at nodes: 731

New cenário? (y/n)

n

Process finished with exit code 0



Algoritmo de destaque

Escolhemos destacar as mudanças que fizemos aos algoritmos dos cenários 2.1 - 2.3 para suportar grafos com arestas repetidas (várias arestas com os mesmos nós de começo e chegada). Este baseia-se no facto de ao ser inserida uma aresta no grafo, ser inserida também, imediatamente a seguir, outra aresta no sentido contrário com capacidade 0. Como isto se verifica, sabemos que a ordem das arestas contrárias é a mesma para as arestas originais. Assim, é possível fazer os acertos de fluxo na rede residual, usando as arestas corretas.



Principais dificuldades, esforço de cada um

A principal dificuldade que sentimos neste trabalho foi no cenário 1.2, pois no início estávamos com problemas em conseguir descobrir como encontrar as soluções ótimas com as restrições do problema.

David Ferreira- $\frac{1}{3}$

Dinis Sousa- $\frac{1}{3}$

Francisca Silva - $\frac{1}{3}$