Universidade do Minho

Mestrado Integrado em Engenharia Informática
Licenciatura em Engenharia Informática

Investigação Operacional - Trabalho Prático #2
Ano Letivo 2021/2022
Grupo 31

Gonçalo Braz (a93178) Simão Cunha (a93262)
Tiago Silva (a93277) Gonçalo Pereira (a93168)

7 de maio de 2022

1 Introdução

Este segundo trabalho prático surge no âmbito da UC de Investigação Operacional. Consiste na distribuição geográfica de clientes a equipas, de modo a minimizar o custo total da operação, que inclui custos de deslocação e custos fixos de utilização de veículos, tendo em conta que estes clientes têm momentos fixos para serem servidos, sendo portanto este um problema do tipo *fixed scheduling*. Será ignorado o tempo de serviço que uma equipa está a prestar a um cliente, pelo que o tempo, nessa situação, toma o valor t = *0*. Uma equipa começará o serviço às 09:00 h na sede da empresa, em Keleirós, irá atender os clientes devidos e deverá regressar a este ponto no final do serviço. O custo fixo de operação de uma equipa é de 1 u.m.

Sabendo que o maior número de aluno do nosso grupo é `93277`, surgiu, assim, a seguinte tabela que lista todos os clientes que serão passíveis de serem atendidos, bem como o seu horário de serviço:

j	**Cliente**	a_j ($\frac{1}{4}$)	a_j **(hora do serviço)**
1	Ana	4	10:00
2	Beatriz	7	10:45
3	Carlos	4	10:00
4	Diogo	2	09:30
5	Eduardo	10	11:30
6	Francisca	6	10:30
7	Gonçalo	9	11:15
8	Helena	3	09:45
9	Inês	2	09:30
10	José	5	10:15

Tabela 1: Listagem de clientes, com horários de serviço em horas e em quartos de hora

2 Formulação do modelo

O problema com que nos deparamos pode, como sugerido no próprio enunciado, ser resolvido através da minimização do fluxo de custo numa rede.

Deste modo, procedemos à construção de um grafo para a respetiva rede. Este grafo tem, no entanto, de seguir certas considerações:

1. O seguinte excerto é uma referência à secção 3.1 do documento `Modelos de Investigação Operacional`:

   Seja V o conjunto de vértices e A o conjunto dos arcos, c_{ij} o custo unitário do arco orientado pertencente a A, u_{ij} a capacidade do arco e b_j o valor da oferta ou procura no vértice j pertencente a V, pretende-se determinar o fluxo admissível de custo mínimo, representado por um conjunto de variáveis de decisão x_{ij}.

$$\min \quad \sum_{(i,j)\in A} c_{ij} x_{ij}$$

$$\text{suj. a} \quad -\sum_{(i,j)\in A} x_{ij} + \sum_{(j,i)\in A} x_{ji} = b_j, \ \forall j \in V \tag{3.1}$$

$$0 \le x_{ij} \le u_{ij}, \ \forall (i,j) \in A \tag{3.2}$$

Figura 1: Restrições da rede

   O grafo deve estar sujeito às restrições de conservação de fluxo (3.1), que força o fluxo que entra num vértice a ser igual ao fluxo que sai, e de capacidade (3.2), que não permite que o fluxo num dado arco exceda a sua capacidade.

2. Os arcos deste grafo serão orientados e terão um custo unitário correspondente ao custo entre o vértice de origem e destino. Além disso, terão ainda um valor referente à sua capacidade.

3. Este custo tem-se através do quadro de custos de deslocação fornecido no enunciado:

	B	C	D	E	F	G	H	I	J	K
A	13	5	6	5	10	7	5	0	7	1
B		11	14	10	8	6	11	13	4	15
C			8	6	10	6	0	5	6	2
D				4	8	8	8	6	11	4
E					6	4	6	5	7	6
F						5	10	10	8	11
G							10	7	5	9
H								5	6	9
I									7	9
J										10

custos de deslocação

Figura 2: Quadro de custos de deslocação

Como utilizamos a matriz originalmente fornecida, alguns dos custos das arestas não seguem a propriedade de desigualdade triangular ($d_{AC} <= d_{AB} + d_{BC}$), ou seja, nem sempre o arco direto entre dois vértices tem um valor menos ou igual à soma dos arcos de um caminho que parte do primeiro vértice e, indiretamente, tem como destino o segundo vértice.

4. Os vértices "interiores"do grafo corresponderão à localização dos diferentes clientes e terão um valor referente à sua procura, que neste caso será -1, visto que um cliente deve apenas ser atendido uma única vez.

5. O grafo terá um vértice que servirá de partida, neste caso será a sede da empresa, e um vértice de chegada que será também a sede da empresa. Todas os caminhos gerados através das soluções calculadas terão de ter início no ponto de partida e fim no ponto de chegada para serem admissíveis.

6. A soma das ofertas tem de ser igual à soma das procuras.

Seguindo o cenário do problema, qualquer cliente terá um caminho que permite a chegada a um qualquer outro cliente. No entanto, os serviços que uma equipa pode realizar estão limitados ao seguinte invariante:

$$a_i + t_{ij} \leq a_j \tag{1}$$

onde:

- a_i é o instante de término de serviço no cliente i;
- a_j é o instante de início do serviço no cliente j;
- t_{ij} é o tempo necessário de ir do cliente i para o cliente j.

Isto significa que um outro cliente apenas poderá ser servido por uma equipa caso a hora de serviço do cliente atual, somada ao tempo de deslocação a esse cliente, resulte num valor igual ou inferior à hora de serviço desse cliente. Além disso, a hora de chegada de uma equipa a um cliente não influencia a hora do serviço, visto que esta é fixa.

Assim, o grafo que podemos construir é limitado por esta restrição.

O primeiro passo a tomar para a resolução do problema foi, então, a elaboração do grafo de compatibilidades, através do seu respetivo quadro, calculado através de uma folha de excel:

	A	B	C	D	E	F	G	H	I	J	K
A					5		7				1
B					10		6				15
C		11			6		6			6	2
D	6	14			4	8	8			11	4
E											6
F					6		5				11
G					4						9
H	5	11	0		6	10	10			6	9
I	0	13	5		5	10	7	5		7	9
J		4			7		5				10
K	2	16	3	5	7	12	10	10	10	11	

Figura 3: Quadro de compatibilidades

A partir dos resultados obtidos do quadro podemos formular o grafo com todas as conexões admissíveis entre os diferentes clientes e respetivos custos, respeitando a condição referida anteriormente.

Além disso, nota-se que os custos na linha da sede da empresa diferem em 1 valor em cada arco do proposto em 2, visto que aplicamos já o custo fixo associado à operação de uma equipa, que é de 1 U.M..

Depois da elaboração do quadro, surgiu o seguinte grafo:

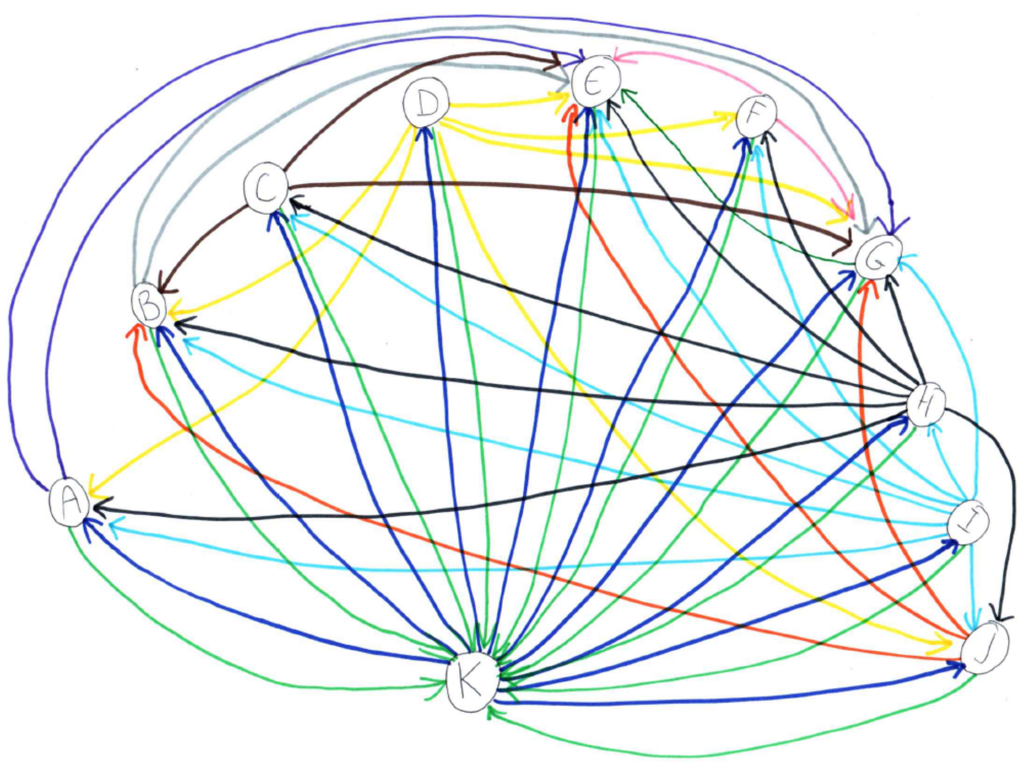


Figura 4: Grafo de compatibilidades

Embora a representação seja um bocado crude, permite ainda assim perceber as deslocações permitidas de uma equipa entre os diferentes clientes.

A partir deste ponto, tivemos de criar uma forma de limitar a capacidade dos vértices em si, visto que estes têm uma capacidade limitada de atendimento. Para isto, recorremos ao desdobramento de todos os vértices, seguindo o princípio enunciado na secção 3.13.2 no documento `Modelos de Investigação Operacional`.

Realizado isto, obtemos o seguinte grafo:

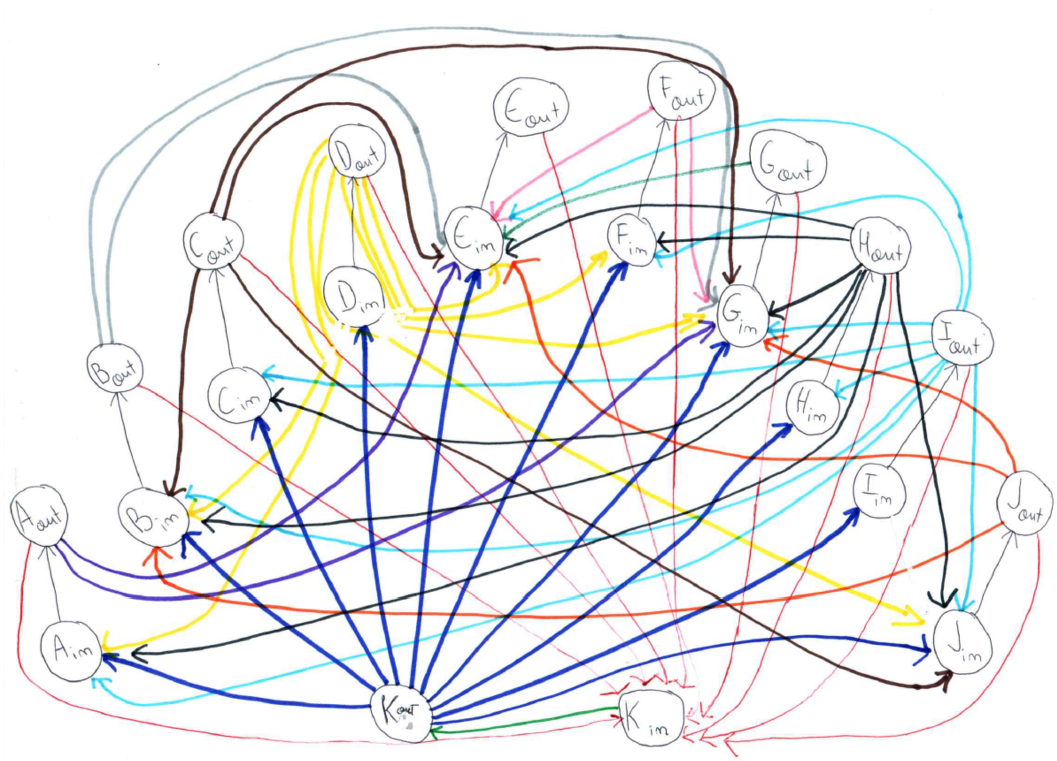


Figura 5: Grafo de compatibilidades desdobrado

Além do desdobramento simples, de modo a permitir várias iterações pelo grafo (várias equipas), no algoritmo, criamos um arco entre o k_{out} e k_{in} com custo 0.

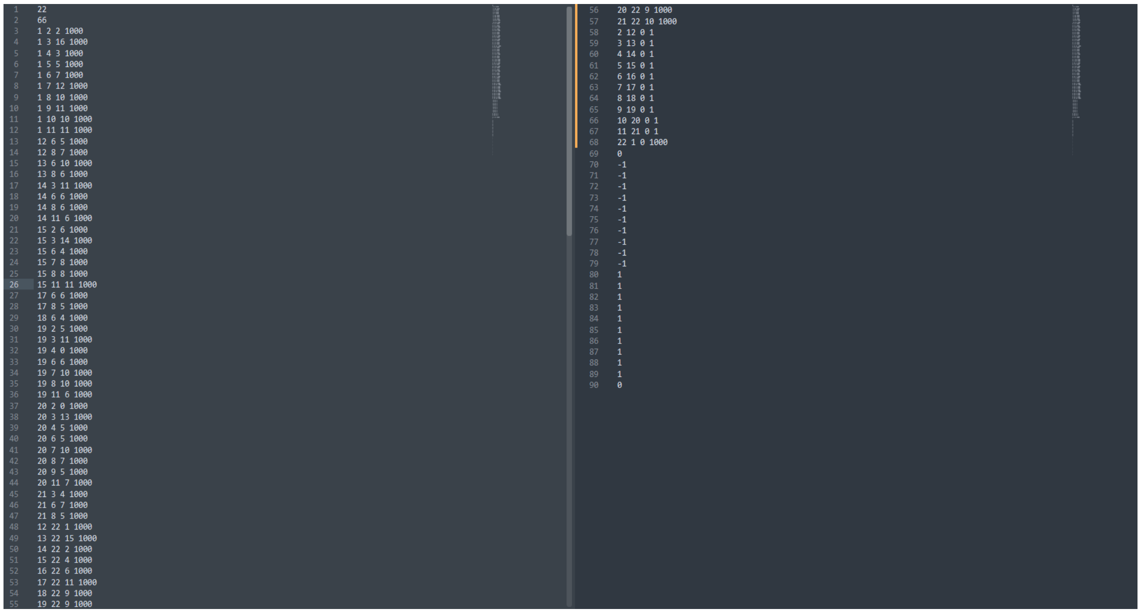
Os novos arcos criados terão custo 0.

Visto que estes arcos resultantes do desdobramento derivam de vértices com procura, estes terão um limite de fluxo inferior igual a esse valor. Para tratarmos desse limite, teremos de aumentar o valor da procura do vértice *in* no número de unidades do limite, e a oferta do vértice *out* da mesma forma.

Neste resultado, além de permitir limitar a capacidade dos vértices, podemos ainda permitir uma continuação do fluxo no decorrer do grafo, sendo que os vértices `in`, excluindo K, terão valor de procura -1 e, os vértices `out`, excluindo novamente o K, terão valor de oferta 1. Assim, garantimos que a quantidade de oferta é igual à quantidade de procura.

Obtidas estas conclusões, podemos passar ao cálculo da solução ótima do modelo. Para isto, utilizaremos o *software* de otimização em rede `Relax4`.

3 Ficheiro de *input*



```
22
66
1 2 2 1000
1 3 16 1000
1 4 3 1000
1 5 5 1000
1 6 7 1000
1 7 12 1000
1 8 10 1000
1 9 11 1000
1 10 10 1000
1 11 11 1000
12 6 5 1000
12 8 7 1000
13 6 10 1000
13 8 6 1000
14 3 11 1000
14 6 6 1000
14 8 6 1000
14 11 6 1000
15 2 6 1000
15 3 14 1000
15 6 4 1000
15 7 8 1000
15 8 8 1000
15 11 11 1000
17 6 6 1000
17 8 5 1000
18 6 4 1000
19 2 5 1000
19 3 11 1000
19 4 0 1000
19 6 6 1000
19 7 10 1000
19 8 10 1000
19 11 6 1000
20 2 0 1000
20 3 13 1000
20 4 5 1000
20 6 5 1000
20 7 10 1000
20 8 7 1000
20 9 5 1000
20 11 7 1000
21 3 4 1000
21 6 7 1000
21 8 5 1000
12 22 1 1000
13 22 15 1000
14 22 2 1000
15 22 4 1000
16 22 6 1000
17 22 11 1000
18 22 9 1000
19 22 9 1000
20 22 9 1000
21 22 10 1000
2 12 0 1
3 13 0 1
4 14 0 1
5 15 0 1
6 16 0 1
7 17 0 1
8 18 0 1
9 19 0 1
10 20 0 1
11 21 0 1
22 1 0 1000
0
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
1
1
1
1
1
1
1
1
1
1
0
```

Figura 6: Ficheiro de input no *software* `Sublime Text`

Vértice	**Denominação Relax4**	**Vértice**	**Denominação Relax4**
K_{in}	1	A_{out}	12
A_{in}	2	B_{out}	13
B_{in}	3	C_{out}	14
C_{in}	4	D_{out}	15
D_{in}	5	E_{out}	16
E_{in}	6	F_{out}	17
F_{in}	7	G_{out}	18
G_{in}	8	H_{out}	19
H_{in}	9	I_{out}	20
I_{in}	10	J_{out}	21
J_{in}	11	K_{out}	22

Tabela 2: Legenda dos vértices

A estrutura do nosso ficheiro de *input* segue a sintaxe do *solver* `Relax4`:

1. nº de vértices;
2. nº de arcos;
3. descrição dos arcos (nodo de origem → nodo de destino → custo → capacidade);
4. oferta;
5. procura.

Seguindo este formato e os dados anteriormente apresentados sabemos que:

- Existem 22 vértices: A_{in} a K_{in} e A_{out} a K_{out};
- Existem 66 arcos: 55 arcos entre nodos externos (por exemplo, A_{out} e E_{in}) e 11 entre nodos intermédios (por exemplo, $A_{in} \rightarrow A_{out}$);
- Todos os arcos vão ter o custo consoante a tabela de custos de deslocação e a capacidade, entre todos os out e in, exceto no vértice K, vai tomar sempre o valor 1000, uma vez que o fator que limita a quantidade de vezes que um vértice é percorrido é apenas o dos arcos de in para out, resultado do desdobramento dos vértices, que terão por sua vez capacidade 1;
- Na procura, o valor a colocar dos vértices A_{in} a J_{in} será -1, sendo portanto atrativo, de forma a uma equipa se possa deslocar para os clientes e a oferta dos vértices A_{out} a J_{out} será 1, uma vez que terão de abandonar o cliente após terem feito o serviço, permitindo que o fluxo se mantenha positivo e consiga, saindo de um cliente, visitar um seguinte antes de retornar à sede. Em K_{in} e K_{out} tomarão o valor 0, uma vez que é o ponto inicial e final do percurso, respetivamente.

4 Ficheiro de *output*

Depois de elaborado o ficheiro de *input*, será altura de o executarmos no *solver* `Relax4`, obtendo o seguinte output:

```
END OF READING
NUMBER OF NODES = 22, NUMBER OF ARCS = 66
CONSTRUCT LINKED LISTS FOR THE PROBLEM
CALLING RELAX4 TO SOLVE THE PROBLEM
***********************************
TOTAL SOLUTION TIME =  0. SECS.
TIME IN INITIALIZATION =  0. SECS.
  1 2  1.
  1 5  1.
  1 10  1.
  13 8  1.
  14 11  1.
  15 7  1.
  17 6  1.
  19 4  1.
  20 9  1.
  21 3  1.
  12 22  1.
  16 22  1.
  18 22  1.
  22 1  3.
OPTIMAL COST =   68.
NUMBER OF AUCTION/SHORTEST PATH ITERATIONS = 45
NUMBER OF ITERATIONS =  36
NUMBER OF MULTINODE ITERATIONS =  6
NUMBER OF MULTINODE ASCENT STEPS =  2
NUMBER OF REGULAR AUGMENTATIONS =  6
***********************************
```

Figura 7: Output obtido no `Relax4`

5 Interpretação e validação da solução

Através do output, conseguimos elaborar uma rede mais simplificada para observarmos quantas equipas serão necessárias para atender os clientes e que equipas atenderão quais clientes.

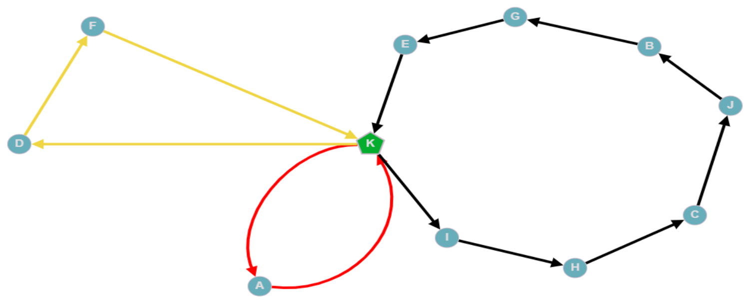


Figura 8: Rede obtida através da interpretação do *output*

Assim, podemos retirar algumas conclusões:

- O custo de deslocamento de toda a operação será de 68 u.m.;
- Serão necessárias 3 equipas: E_1, E_2 e E_3;
- A equipa E_1 parte da sede e vai atender o cliente A e regressa à sede;
- A equipa E_2 parte da sede e irá atender os clientes D e F, regressando a Keleirós;
- A equipa E_3 parte de Keleirós e vai deslocar-se para os pontos I, H, C, J, B, G e E, regressando à sede da sua empresa.

Verificamos então, que a solução obtida é admissível visto que todas as equipas têm ponto de partida na sede da empresa e acabam o seu percurso na mesma. Além disso, pelos cálculos efetuados anteriormente no quadro de compatibilidades, podemos afirmar que os arcos percorridos nos caminhos são todos válidos, visto que estes foram retirados desse quadro, sendo portanto o caminho em si válido no problema de fixed scheduling em questão.

6 Conclusão

Neste trabalho prático, dedicamo-nos à consolidação de conhecimentos obtidos nas aulas teóricas e práticas, nomeadamente a vertente do problema de transportes: redes com capacidade. Conseguimos ganhar experiência na utilização de *software* de otimização de redes como o `Relax4`, crendo que obtivemos a melhor solução de forma a minimizar o custo total da distribuição de clientes a equipas. Além disso, foi possível adaptarmos um problema de *fixed scheduling* numa rede com capacidades e tivemos a necessidade de aplicar as devidas transformações ao grafo de compatibilidades obtido de modo a cumprir as restrições sobre o fluxo em redes.