

UNIVERSIDADE DO MINHO

DEPARTAMENTO DE INFORMÁTICA

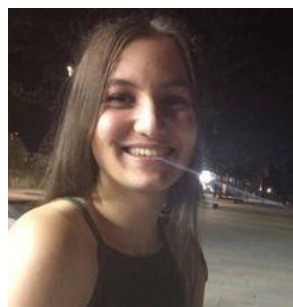
TRABALHO PRÁTICO 1

DRONE

Modelos Determinísticos de Investigação Operacional
13 de novembro de 2020



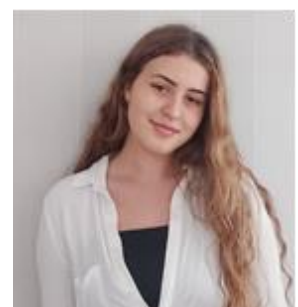
(a) Ana Catarina Canelas
A93872



(b) Ana Filipa Pereira
A89589



(c) Carolina Santejo
A89500



(d) Raquel Costa
A89464

Conteúdo

Introdução.....	3
Objetivo Principal:.....	3
Grafo do Grupo.....	4
Formulação do Problema.....	4
Variáveis de decisão.....	5
Função Objetivo.....	5
Restrições.....	5
Modelo de Programação Linear.....	6
Variáveis de decisão.....	6
Parâmetros.....	6
Função Objetivo.....	7
Restrições.....	7
LP Solve.....	8
Ficheiro de <i>Input</i>	8
Ficheiro de <i>Output</i>	9
Solução Ótima.....	10
Análise Final.....	12
Interpretação da solução ótima.....	12
Validação do Modelo.....	12
Reflexão Crítica.....	14
Conclusão.....	17
Figura 1 - Grafo Inicial.....	3
Figura 2 - Grafo Resultante.....	4
Figura 3 - Input dado no LPSolve.....	8
Figura 4 - Output do LPSolve.....	9
Figura 5 - Grafo Final.....	10
Figura 6 - Possibilidade de Percurso.....	11
Figura 7 - Resultado das Restrições.....	12
Figura 8 - Tabelas de Distâncias Euclidianas entre os vértices.....	13
Figura 9 - Variáveis do Modelo Anterior.....	14
Figura 10 - Função Objetivo do Modelo Anterior.....	14
Figura 11 - 1ª Restrição do Modelo Anterior.....	15
Figura 12 - 2ª Restrição do Modelo Anterior.....	15
Figura 13 - Resultado do Modelo Anterior.....	16

Introdução

No âmbito da Unidade Curricular de Modelos Determinísticos de Investigação Operacional, desenvolvemos o 1º Trabalho Prático proposto para este ano Ano Letivo. Neste trabalho abordamos o problema do “Drone”, sendo este bastante semelhante ao problema do Carteiro Chinês. O cenário apresentado consiste na inspeção de todas as linhas de alta tensão pelo drone de modo a verificar se há vegetação a interferir com as mesmas. Para que o drone consiga realocar-se para fazer a inspeção de uma nova linha não é necessário seguir as linhas de alta tensão, bastando fazer o percurso mais curto através do ar. É fornecida também uma tabela com as distâncias Euclidianas necessárias à resolução deste desafio.

Na resolução deste problema, recorreremos ao software de programação linear *LPSolve*.

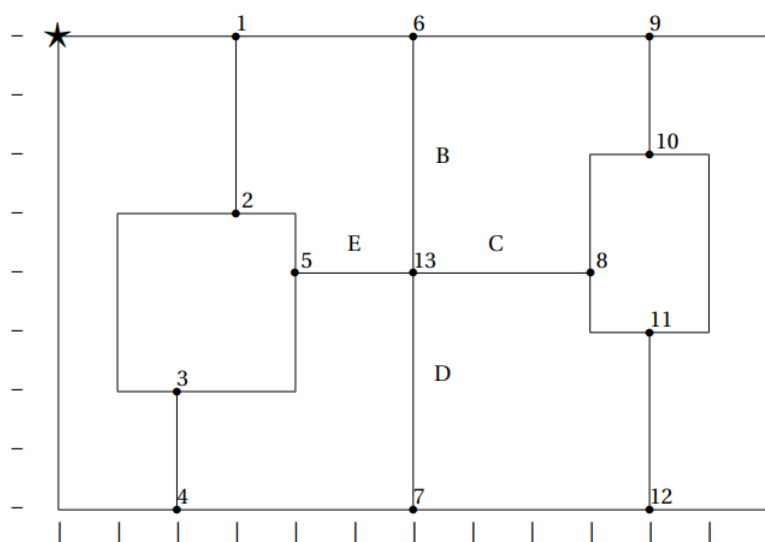


Figura 1 - Grafo Inicial

Objetivo Principal:

O principal objetivo trata-se de encontrar a solução ótima para o problema. Para tal temos de encontrar um caminho que comece e acabe no ponto inicial ao qual chamamos de “ponto estrela”, e, além disso, calcular a distância mínima percorrida pelo drone, sendo que este tem de passar por todas as linhas de alta tensão (correspondem a arestas do grafo - Grafo Inicial) pelo menos uma vez. É importante referir também que cada aresta/linha pode ser percorrida mais de uma vez, e em qualquer sentido.

emparelhamento é uma enumeração de pares formados por vértices de grau ímpar que traduzem uma possibilidade de arestas a adicionar ao grafo de forma a que este passe a ter um circuito *Euleriano*. Desta forma, podemos perceber que o emparelhamento ótimo é aquele, que entre todas as opções, representa o conjunto de arestas “extra” a percorrer com menor custo.

Variáveis de decisão

Como já foi referido, o nosso grafo tem 12 vértices de grau ímpar. Dado que queremos ter em conta todas as possibilidades de pares entre esses mesmos vértices de grau ímpar, o modelo de programação linear terá 66 variáveis de decisão. Estas V.D são do tipo X_{ij} que representa o par formado pelos vértices i e j (vértices de grau ímpar), sendo que estas variáveis são binárias, uma vez que o objetivo é saber se o par estará ou não no emparelhamento ótimo. Na realidade, cada variável de decisão traduz uma possibilidade de aresta(s) a adicionar ao grafo de forma a tornar os vértices i e j de grau par. Além disto, a escolha das arestas depende do seu custo logo os coeficientes de cada uma das V.D corresponde á menor distância entre os vértices que formam o par.

Função Objetivo

Visto que queremos encontrar o emparelhamento ótimo, a função objetivo será obrigatoriamente de minimização (minimiza o custo dos pares de vértices a escolher) e dará como resultado o custo total mínimo dos caminhos “extra” a percorrer pelo *drone*.

Restrições

Temos de garantir que o emparelhamento escolhido não contém pares diferentes com vértices em comum. Caso isto acontecesse, verificar-se-ia uma de duas situações: ou o vértice torna-se par, mas foram adicionadas arestas desnecessárias que representam um aumento distância total percorrida, ou então o vértice poderá continuar a ser de grau ímpar, não resolvendo o problema inicial. Deste modo, foi necessária uma restrição para cada vértice de grau ímpar, que impede que se selecionem duas variáveis de decisão com valores de índice em comum.

Modelo de Programação Linear

Variáveis de decisão

Como já foi referido, X_{ij} traduz uma possibilidade de aresta(s) a adicionar ao grafo, no processo de o tornar *Euleriano*. Visto que os pares são formados apenas pelos vértices de grau ímpar, i e j têm de pertencer a um grupo restrito. Além disto não é permitido ter $i = j$ pois isso não representaria nenhuma aresta. Também não podemos ter variáveis que representem o mesmo par (X_{12} e X_{21}), daí consideramos $j > i$ (poderia ser $i < j$ mas no nosso trabalho escolhemos a primeira opção)

$$X_{ij} \in \{0, 1\}, \quad \forall i, j \in \{1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13\}, \quad i \neq j \wedge j > i$$

Parâmetros

Os parâmetros são os dados do sistema que não podem ser alterados. Neste problema em específico é perceptível que os parâmetros são:

- A distância entre dois vértices e consequentemente C_{ij} , que traduz a distância *Euclidiana* entre os vértices i e j (que poderá coincidir com uma linha de alta tensão).

Função Objetivo

A função objetivo é de minimização pois queremos encontrar o emparelhamento de menor custo.

$$\min z = \sum_{i=1}^5 \sum_{j=i+1}^6 C_{ij} * X_{ij} + \sum_{i=1}^6 \sum_{j=8}^{13} C_{ij} * X_{ij} + \sum_{i=8}^{12} \sum_{j=i+1}^{13} C_{ij} * X_{ij}$$

Restrições

As restrições garantem que o emparelhamento é formado apenas por pares válidos, ou seja, como foi explicado anteriormente, temos de garantir que o vértice v (de grau ímpar) está incluído em um e um só par desse mesmo emparelhamento ótimo. Na restrição abaixo, não temos em conta a ordem dos vértices no emparelhamento (por exemplo, temos X_{21} ao invés de X_{12} , mas é indiferente pois representam o mesmo par).

Sejam $V_1 = \{1,2,3,4,5,6,8,9,10,11,12,13\}$

Sujeito a $\sum_{j \in V \setminus \{i\}} X_{ij} = 1, \forall i \in \{1,2,3,4,5,6,8,9,10,11,12\}$

LP Solve

Ficheiro de *Input*

```
1 /* Objective function */
2 // xij -> emparelhamento. = caso não seja escolhido. 1 Caso seja.
3
4 min: 3 x12 + 6.08 x13 + 8.06 x14 + 4.12 x15 + 3 x16 + 7.21 x18 + 7 x19 + 7.28 x1_10 + 8.60 x1_11 + 10.63 x1_12 + 5 x1_13 + 3.16 x23
5 + 5.10 x24 + 1.41 x25 + 4.24 x26 + 6.08 x28 + 7.62 x29 + 7.07 x2_10 + 7.28 x2_11 + 8.60 x2_12 + 3.16 x2_13 + 2 x34 + 2.83 x35
6 + 7.21 x36 + 7.28 x38 + 10 x39 + 8.94 x3_10 + 8.06 x3_11 + 8.25 x3_12 + 4.47 x3_13 + 4.47 x45 + 8.94 x46 + 8.06 x48 + 11.31 x49
7 + 10 x4_10 + 8.54 x4_11 + 8 x4_12 + 5.66 x4_13 + 4.47 x56 + 5 x58 + 7.21 x59 + 6.32 x5_10 + 6.08 x5_11 + 7.21 x5_12 + 2 x5_13
8 + 5 x68 + 4 x69 + 4.47 x6_10 + 6.40 x6_11 + 8.94 x6_12 + 4 x6_13 + 4.12 x89 + 2.24 x8_10 + 1.41 x8_11 + 4.12 x8_12 + 3 x8_13
9 + 2 x9_10 + 5 x9_11 + 8 x9_12 + 5.66 x9_13 + 3 x10_11 + 6 x10_12 + 4.47 x10_13 + 3 x11_12 + 4.12 x11_13 + 5.66 x12_13 ;
10
11 /* Variable bounds */
12 /*Garantir que as parelhas são válidas*/
13 Vertice1: x12 + x13 + x14 + x15 + x16 + x18 + x19 + x1_10 + x1_11 + x1_12 + x1_13 = 1 ;
14 Vertice2: x12 + x23 + x24 + x25 + x26 + x28 + x29 + x2_10 + x2_11 + x2_12 + x2_13 = 1 ;
15 Vertice3: x13 + x23 + x34 + x35 + x36 + x38 + x39 + x3_10 + x3_11 + x3_12 + x3_13 = 1 ;
16 Vertice4: x14 + x24 + x34 + x45 + x46 + x48 + x49 + x4_10 + x4_11 + x4_12 + x4_13 = 1 ;
17 Vertice5: x15 + x25 + x35 + x45 + x56 + x58 + x59 + x5_10 + x5_11 + x5_12 + x5_13 = 1 ;
18 Vertice6: x16 + x26 + x36 + x46 + x56 + x68 + x69 + x6_10 + x6_11 + x6_12 + x6_13 = 1 ;
19 Vertice8: x18 + x28 + x38 + x48 + x58 + x68 + x89 + x8_10 + x8_11 + x8_12 + x8_13 = 1 ;
20 Vertice9: x19 + x29 + x39 + x49 + x59 + x69 + x89 + x9_10 + x9_11 + x9_12 + x9_13 = 1 ;
21 Vertice10: x1_10 + x2_10 + x3_10 + x4_10 + x5_10 + x6_10 + x8_10 + x9_10 + x10_11 + x10_12 + x10_13 = 1 ;
22 Vertice11: x1_11 + x2_11 + x3_11 + x4_11 + x5_11 + x6_11 + x8_11 + x9_11 + x10_11 + x11_12 + x11_13 = 1 ;
23 Vertice12: x1_12 + x2_12 + x3_12 + x4_12 + x5_12 + x6_12 + x8_12 + x9_12 + x10_12 + x11_12 + x12_13 = 1 ;
24 Vertice13: x1_13 + x2_13 + x3_13 + x4_13 + x5_13 + x6_13 + x8_13 + x9_13 + x10_13 + x11_13 + x12_13 = 1 ;
25
26
27 bin x12 , x13 , x14 , x15 , x16 , x18 , x19 , x1_10 , x1_11 , x1_12 , x1_13 ;
28 bin x23 , x24 , x25 , x26 , x28 , x29 , x2_10 , x2_11 , x2_12 , x2_13 ;
29 bin x34 , x35 , x36 , x38 , x39 , x3_10 , x3_11 , x3_12 , x3_13 ;
30 bin x45 , x46 , x48 , x49 , x4_10 , x4_11 , x4_12 , x4_13 ;
31 bin x56 , x58 , x59 , x5_10 , x5_11 , x5_12 , x5_13 ;
32 bin x68 , x69 , x6_10 , x6_11 , x6_12 , x6_13 ;
33 bin x89 , x8_10 , x8_11 , x8_12 , x8_13 ;
34 bin x9_10 , x9_11 , x9_12 , x9_13 ;
35 bin x10_11 , x10_12 , x10_13 ;
36 bin x11_12 , x11_13 ;
37 bin x12_13 ;
```

Figura 3 - Input dado no LPSolve

Ficheiro de *Output*

Source			Matrix			Options			Result		
Objective			Constraints			Sensitivity					
Variables		MILP ...	result								
		14,41	14,41								
x9_10		1	1								
x8_13		1	1								
x34		1	1								
x25		1	1		x3_12		0	0			
x16		1	1		x3_11		0	0			
x11_12		1	1		x3_10		0	0			
x9_13		0	0		x39		0	0			
x9_12		0	0		x38		0	0			
x9_11		0	0		x36		0	0			
x8_12		0	0		x35		0	0			
x8_11		0	0		x2_13		0	0			
x8_10		0	0		x2_12		0	0			
x89		0	0		x2_11		0	0			
x6_13		0	0		x2_10		0	0			
x6_12		0	0		x29		0	0			
x6_11		0	0		x28		0	0			
x6_10		0	0		x26		0	0			
x69		0	0		x24		0	0			
x68		0	0		x23		0	0			
x5_13		0	0		x1_13		0	0			
x5_12		0	0		x1_12		0	0			
x5_11		0	0		x1_11		0	0			
x5_10		0	0		x1_10		0	0			
x59		0	0		x19		0	0			
x58		0	0		x18		0	0			
x56		0	0		x15		0	0			
x4_13		0	0		x14		0	0			
x4_12		0	0		x13		0	0			
x4_11		0	0		x12_13		0	0			
x4_10		0	0		x12		0	0			
x49		0	0		x11_13		0	0			
x48		0	0		x10_13		0	0			
x46		0	0		x10_12		0	0			
x45		0	0		x10_11		0	0			
x3_13		0	0								

Figura 4 - Output do LPSolve

A solução ótima é o emparelhamento de cardinal 6 correspondente a $\{(1,6), (2,5), (3,4), (8,13), (9,10), (11,12)\}$ e o custo total do emparelhamento ótimo é 14.41 (unidades de distância).

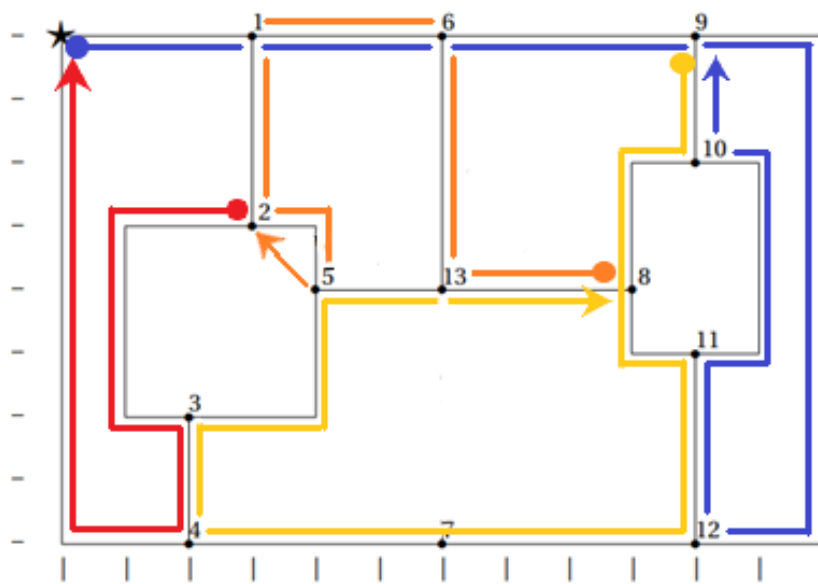


Figura 6 - Possibilidade de Percurso

Percurso:

Roxo → **Amarelo** → **Laranja** → **Vermelho**;

Isto é:

★ → 1 → 6 → 9 → 12 → 11 → 10 → 9 → 10 → 8 → 11 → 12 → 7 → 4 → 3
 → 5 → 13 → 8 → 13 → 6 → 1 → 2 → 5 → 2 → 3 → 4 → ★

Sabendo que a soma da distância de cada aresta do grafo inicial é:

$$8 + 8 + 12 + 12 + 2 + 3 + 3 + 3 + 3 + 3 + 3 + 2 + 4 + 3 + 3 + 2 + 2 + 5 = 81$$

E que o resultado obtido no LP Solver é 14.41. Então a distância total percorrida é:

$$81 + 14.41 = \mathbf{95.41}$$

Análise Final

Interpretação da solução ótima

Como se pode verificar na Figura 4, foram escolhidos os valores das variáveis de decisão pelo LP Solve, para que o custo total do emparelhamento seja mínimo. Existem seis variáveis com valor 1, isto significa que cada uma delas corresponde, á duplicação da(s) aresta(s) que compõem o caminho mais curto entre os vértices de cada par, no grafo inicial para o tornar Euleriano. As que têm valor 0 não são incluídas pois não correspondem à solução ótima. Neste caso concreto, o caminho mais curto entre cada um dos vértices de cada par é composto por apenas uma aresta (seja ela linha de tensão ou aresta diagonal).

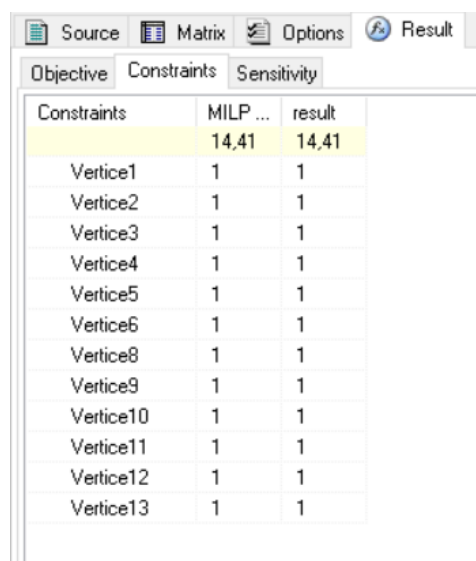
Validação do Modelo

Após calcular o resultado ótimo no *LP Solve* foi necessário verificar que a solução obtida satisfaz todas a condições idealizadas inicialmente e se adequa ao sistema na realidade. Assim, para validar o modelo foram efetuadas as seguintes verificações:

- Nenhuma distância é negativa ou nula (Figura 8);

$$distancia\ total\ percorrida > 0 \equiv 95.41 > 0$$

- Os emparelhamentos escolhidos não têm vértices repetidos, ou seja, para cada vértice foi escolhido apenas um arco (Figura 7);



Objective	Constraints	Sensitivity
Constraints	MILP ...	result
	14,41	14,41
Vertice1	1	1
Vertice2	1	1
Vertice3	1	1
Vertice4	1	1
Vertice5	1	1
Vertice6	1	1
Vertice8	1	1
Vertice9	1	1
Vertice10	1	1
Vertice11	1	1
Vertice12	1	1
Vertice13	1	1

Figura 7 - Resultado das Restrições

- Todos os vértices do grafo têm grau par (Figura 5);
- É possível efetuar um percurso (com origem e destino no vértice ★) que passa em todas as arestas necessárias apenas uma vez, acrescentado unicamente os emparelhamentos selecionados (solução ótima – Figura 6);
- O valor da distância total percorrida é o mínimo possível de acordo com todas as restrições. De acordo com o percurso escolhido temos:

$$\begin{aligned} \text{distancia percorrida} = & 3 + 3 + 4 + 2 + 8 + 2 + 3 + 1 + 3 + 1 + 2 + 2 + 1 + 3 + \\ & 1 + 3 + 4 + 4 + 2 + 2 + 2 + 2 + 3 + 3 + 4 + 3 + 3 + 1 + 1 + 1.41 + 2 + 3 + \\ & 1 + 2 + 2 + 8 = 95.41 \equiv \text{solução ótima} = 95.41 \end{aligned}$$

- A distância total percorrida pelo drone é admissível (o valor resultante é realista), por exemplo, o seu valor é maior do que a soma do comprimento de todas as arestas do grafo inicial (arestas que o drone é obrigado a percorrer).

$$\text{distancia total percorrida} > \text{soma arestas grafo inicial} \equiv 95.41 > 81$$

		x	3	3	2	2	4	6	6	9	10	10	10	10	6
		y	8	5	2	0	4	8	0	4	8	6	3	0	4
			1	2	3	4	5	6	7	8	9	10	11	12	13
3	8	1	0,00	3,00	6,08	8,06	4,12	3,00	8,54	7,21	7,00	7,28	8,60	10,63	5,00
3	5	2	3,00	0,00	3,16	5,10	1,41	4,24	5,83	6,08	7,62	7,07	7,28	8,60	3,16
2	2	3	6,08	3,16	0,00	2,00	2,83	7,21	4,47	7,28	10,00	8,94	8,06	8,25	4,47
2	0	4	8,06	5,10	2,00	0,00	4,47	8,94	4,00	8,06	11,31	10,00	8,54	8,00	5,66
4	4	5	4,12	1,41	2,83	4,47	0,00	4,47	4,47	5,00	7,21	6,32	6,08	7,21	2,00
6	8	6	3,00	4,24	7,21	8,94	4,47	0,00	8,00	5,00	4,00	4,47	6,40	8,94	4,00
6	0	7	8,54	5,83	4,47	4,00	4,47	8,00	0,00	5,00	8,94	7,21	5,00	4,00	4,00
9	4	8	7,21	6,08	7,28	8,06	5,00	5,00	5,00	0,00	4,12	2,24	1,41	4,12	3,00
10	8	9	7,00	7,62	10,00	11,31	7,21	4,00	8,94	4,12	0,00	2,00	5,00	8,00	5,66
10	6	10	7,28	7,07	8,94	10,00	6,32	4,47	7,21	2,24	2,00	0,00	3,00	6,00	4,47
10	3	11	8,60	7,28	8,06	8,54	6,08	6,40	5,00	1,41	5,00	3,00	0,00	3,00	4,12
10	0	12	10,63	8,60	8,25	8,00	7,21	8,94	4,00	4,12	8,00	6,00	3,00	0,00	5,66
6	4	13	5,00	3,16	4,47	5,66	2,00	4,00	4,00	3,00	5,66	4,47	4,12	5,66	0,00

Figura 8 - Tabelas de Distâncias Euclidianas entre os vértices

Reflexão Crítica

A abordagem realizada para tratar o problema em causa foi escolhida após ter sido feita uma comparação com outra resolução encontrada pelo grupo previamente. Com o decorrer do projeto, apercebemo-nos de que este primeiro modelo encontrado não se tratava de uma solução eficiente. Em primeiro lugar, esta solução abarcava 180 variáveis do tipo X_{ijk} representando o número de vezes que cada aresta entre um par de vértices é percorrida, sendo i e j os vértices da aresta em causa e k seria “A” se a aresta fosse aérea ou “L” caso se tratasse de uma linha de tensão (Figura 9). Além disso foram considerados os dois sentidos em que a aresta pode ser percorrida e também o vértice 7. Sendo isto desnecessário, uma vez que o 7 é um ponto que apenas pertence à aresta entre os vértices 4 e 12 do grafo do grupo.

```
int x01L , x10L , x02A , x20A , x03A , x30A , x04A , x40A ;
int x04L , x40L , x05A , x50A , x07A , x70A , x08A , x80A ;
int x0_10A , x10_0A , x0_11A , x11_0A , x0_12A , x12_0A ;
int x0_13A , x13_0A , x12L , x21L , x16L , x61L , x13A , x31A ;
int x14A , x41A , x15A , x51A , x17A , x71A , x18A , x81A ;
int x1_10A , x10_1A , x1_11A , x11_1A , x1_12A , x12_1A , x1_13A , x13_1A ;
int x23A , x32A , x23L , x32L , x25A , x52A , x25L , x52L , x24A , x42A ;
int x26A , x62A , x27A , x72A , x28A , x82A , x29A , x92A ;
int x2_10A , x10_2A , x2_11A , x11_2A , x2_12A , x12_2A , x2_13A , x13_2A ;
int x35A , x53A , x35L , x53L , x34L , x43L , x36A , x63A , x37A , x73A ;
int x38A , x83A , x39A , x93A , x3_10A , x10_3A , x3_11A , x11_3A ;
int x3_12A , x12_3A , x3_13A , x13_3A , x47L , x74L , x45A , x54A ;
int x46A , x64A , x48A , x84A , x49A , x94A , x4_10A , x10_4A ;
int x4_11A , x11_4A , x4_12A , x12_4A , x4_13A , x13_4A , x5_13L , x13_5L ;
int x56A , x65A , x57A , x75A , x59A , x95A , x5_10A , x10_5A ;
int x5_11A , x11_5A , x5_12A , x12_5A , x6_13L , x13_6L , x69L , x96L ;
int x67A , x76A , x68A , x86A , x6_10A , x10_6A , x6_11A , x11_6A ;
int x6_12A , x12_6A , x7_12L , x12_7L , x78A , x87A , x79A , x97A ;
int x7_10A , x10_7A , x7_11A , x11_7A , x7_13A , x13_7A , x8_13L , x13_8L ;
int x8_10L , x10_8L , x8_11L , x11_8L , x8_10A , x10_8A , x8_11A , x11_8A ;
int x89A , x98A , x8_12A , x12_8A , x9_10L , x10_9L , x9_12L , x12_9L ;
int x9_13A , x13_9A , x10_11L , x11_10L , x10_11A , x11_10A ;
int x10_13A , x13_10A , x11_12L , x12_11L , x11_13A , x13_11A , x12_13A , x13_12A ;
```

Figura 9 - Variáveis do Modelo Anterior

A função objetivo (Figura 10), tal como a função que foi proposta pelo grupo para a resolução do problema, é de minimização, onde cada variável multiplica pela distância da aresta representada.

```
min: 3 x01L + 3 x10L + 4.24 x02A + 4.24 x20A + 6.32 x03A + 6.32 x30A + 8.25 x04A + 8.25 x40A
+ 10 x04L + 10 x40L + 5.66 x05A + 5.66 x50A + 10 x07A + 10 x70A + 9.85 x08A + 9.85 x80A
+ 10.2 x0_10A + 10.2 x10_0A + 10.3 x0_11A + 10.3 x11_0A + 12.81 x0_12A + 12.81 x12_0A
+ 7.21 x0_13A + 7.21 x13_0A + 3 x12L + 3 x21L + 3 x16L + 3 x61L + 6.08 x13A + 6.08 x31A
+ 8.06 x14A + 8.06 x41A + 4.12 x15A + 4.12 x51A + 8.54 x17A + 8.54 x71A + 7.21 x18A + 7.21 x81A
+ 7.28 x1_10A + 7.28 x10_1A + 8.60 x1_11A + 8.60 x11_1A + 10.63 x1_12A + 10.63 x12_1A + 5 x1_13A + 5 x13_1A
+ 3.16 x23A + 3.16 x32A + 6 x23L + 6 x32L + 1.41 x25A + 1.41 x52A + 2 x25L + 2 x52L + 5.10 x24A + 5.10 x42A
+ 4.24 x26A + 4.24 x62A + 5.83 x27A + 5.83 x72A + 6.08 x28A + 6.08 x82A + 7.62 x29A + 7.62 x92A
+ 7.07 x2_10A + 7.07 x10_2A + 7.28 x2_11A + 7.28 x11_2A + 8.60 x2_12A + 8.60 x12_2A + 3.16 x2_13A + 3.16 x13_2A
+ 2.83 x35A + 2.83 x53A + 4 x35L + 4 x53L + 2 x34L + 2 x43L + 7.21 x36A + 7.21 x63A + 4.47 x37A + 4.47 x73A
+ 7.28 x38A + 7.28 x83A + 10 x39A + 10 x93A + 8.94 x3_10A + 8.94 x10_3A + 8.06 x3_11A + 8.06 x11_3A
+ 8.25 x3_12A + 8.25 x12_3A + 4.47 x3_13A + 4.47 x13_3A + 4 x47L + 4 x74L + 4.47 x45A + 4.47 x54A
+ 8.94 x46A + 8.94 x64A + 8.06 x48A + 8.06 x84A + 11.31 x49A + 11.31 x94A + 10 x4_10A + 10 x10_4A
+ 8.54 x4_11A + 8.54 x11_4A + 5.66 x4_13A + 5.66 x13_4A + 2 x5_13L + 2 x13_5L
+ 4.47 x56A + 4.47 x65A + 4.47 x57A + 4.47 x75A + 7.21 x59A + 7.21 x95A + 6.32 x5_10A + 6.32 x10_5A
+ 6.08 x5_11A + 6.08 x11_5A + 7.21 x5_12A + 7.21 x12_5A + 4 x6_13L + 4 x13_6L + 4 x69L + 4 x96L
+ 5 x68A + 5 x86A + 4.47 x6_10A + 4.47 x10_6A + 6.40 x6_11A + 6.40 x11_6A
+ 8.94 x6_12A + 8.94 x12_6A + 4 x7_12L + 4 x12_7L + 5 x78A + 5 x87A + 8.94 x79A + 8.94 x97A
+ 7.21 x7_10A + 7.21 x10_7A + 5 x7_11A + 5 x11_7A + 4 x7_13A + 4 x13_7A + 3 x8_13L + 3 x13_8L
+ 3 x8_10L + 3 x10_8L + 2 x8_11L + 2 x11_8L + 2.24 x8_10A + 2.24 x10_8A + 1.41 x8_11A + 1.41 x11_8A
+ 4.12 x89A + 4.12 x98A + 4.12 x8_12A + 4.12 x12_8A + 2 x9_10L + 2 x10_9L + 12 x9_12L + 12 x12_9L
+ 5.66 x9_13A + 5.66 x13_9A + 5 x10_11L + 5 x11_10L + 3 x10_11A + 3 x11_10A
+ 4.47 x10_13A + 4.47 x13_10A + 3 x11_12L + 3 x12_11L + 4.12 x11_13A + 4.12 x13_11A + 5.66 x12_13A + 5.66 x13_12A ;
```

Figura 10 - Função Objetivo do Modelo Anterior

Em relação às restrições focamo-nos em dois pontos fulcrais:

- Todas as arestas (linhas de alta tensão) do grafo têm de ser percorridas pelo menos uma vez (Figura 11);
- O número de entradas num vértice, ou seja, o número de arestas convergentes, tem de ser igual ao número de saídas, arestas divergentes (Figura 12). Isto é:

$$X_{10L} = X_{01L} \Leftrightarrow X_{10L} - X_{01L} = 0$$

```

arestaA: x01L + x10L >= 1;
arestaB: x04L + x40L >= 1;
arestaC: x12L + x21L >= 1;
arestaD: x23L + x32L >= 1;
arestaE: x25L + x52L >= 1;
arestaF: x35L + x53L >= 1;
arestaG: x34L + x43L >= 1;
arestaH: x5_13L + x13_5L >= 1;
arestaI: x47L + x74L >= 1;
arestaJ: x16L + x61L >= 1;
arestaK: x6_13L + x13_6L >= 1;
arestaL: x69L + x96L >= 1;
arestaM: x13_8L + x8_13L >= 1;
arestaN: x7_12L + x12_7L >= 1;
arestaO: x9_12L + x12_9L >= 1;
arestaP: x9_10L + x10_9L >= 1;
arestaQ: x11_12L + x12_11L >= 1;
arestaR: x8_10L + x10_8L >= 1;

vertice0: x10L + x40L + x20A + x30A + x40A + x50A + x70A + x80A + x10_0A + x11_0A + x12_0A + x13_0A
- x01L - x04L - x02A - x03A - x04A - x05A - x07A - x08A - x0_10A - x0_11A - x0_12A - x0_13A = 0 ;

vertice1: x61L + x01L + x21L + x31A + x41A + x51A + x71A + x81A + x10_1A + x11_1A + x12_1A + x13_1A
- x16L - x10L - x12L - x13A - x14A - x15A - x17A - x18A - x1_10A - x1_11A - x1_12A - x1_13A = 0 ;

vertice2: x12L + x32L + x52L + x02A + x32A + x52A + x42A + x62A + x72A + x82A + x92A + x10_2A + x11_2A + x12_2A + x13_2A
- x21L - x23L - x25L - x20A - x23A - x25A - x24A - x26A - x27A - x28A - x29A - x2_10A - x2_11A - x2_12A - x2_13A = 0 ;

vertice3: x23L + x43L + x53L + x03A + x13A + x23A + x53A + x63A + x73A + x83A + x93A + x10_3A + x11_3A + x12_3A + x13_3A
- x32L - x34L - x35L - x30A - x31A - x32A - x35A - x36A - x37A - x38A - x39A - x3_10A - x3_11A - x3_12A - x3_13A = 0 ;

vertice4: x04L + x34L + x74L + x04A + x14A + x24A + x54A + x64A + x84A + x94A + x10_4A + x11_4A + x13_4A
- x40L - x43L - x47L - x40A - x41A - x42A - x45A - x46A - x48A - x49A - x4_10A - x4_11A - x4_13A = 0 ;

vertice5: x25L + x35L + x13_5L + x05A + x15A + x25A + x35A + x45A + x65A + x75A + x95A + x10_5A + x11_5A + x12_5A
- x52L - x53L - x5_13L - x50A - x51A - x52A - x53A - x54A - x56A - x57A - x59A - x5_10A - x5_11A - x5_12A = 0 ;

vertice6: x16L + x96L + x13_6L + x26A + x36A + x46A + x56A + x86A + x10_6A + x11_6A + x12_6A
- x61L - x69L - x6_13L - x62A - x63A - x64A - x65A - x68A - x6_10A - x6_11A - x6_12A = 0 ;

vertice7: x47L + x12_7L + x07A + x17A + x27A + x37A + x57A + x87A + x97A + x10_7A + x11_7A + x13_7A
- x74L - x7_12L - x70A - x71A - x72A - x73A - x75A - x78A - x79A - x7_10A - x7_11A - x7_13A = 0 ;

vertice8: x10_8L + x11_8L + x13_8L + x08A + x18A + x28A + x38A + x48A + x68A + x78A + x98A + x10_8A + x11_8A + x12_8A
- x8_10L - x8_11L - x8_13L - x80A - x81A - x82A - x83A - x84A - x86A - x87A - x89A - x8_10A - x8_11A - x8_12A = 0 ;

vertice9: x69L + x10_9L + x12_9L + x29A + x39A + x49A + x59A + x79A + x89A + x13_9A
- x96L - x9_10L - x9_12L - x92A - x93A - x94A - x95A - x97A - x98A - x9_13A = 0 ;

vertice10: x9_10L + x8_10L + x11_10L + x0_10A + x1_10A + x2_10A + x3_10A + x4_10A + x5_10A + x6_10A + x7_10A + x8_10A + x11_10A + x13_10A
- x10_9L - x10_8L - x10_11L - x10_0A - x10_1A - x10_2A - x10_3A - x10_4A - x10_5A - x10_6A - x10_7A - x10_8A - x10_11A - x10_13A = 0 ;

vertice11: x8_11L + x10_11L + x12_11L + x0_11A + x1_11A + x2_11A + x3_11A + x4_11A + x5_11A + x6_11A + x7_11A + x8_11A + x10_11A + x13_11A
- x11_8L - x11_10L - x11_12L - x11_0A - x11_1A - x11_2A - x11_3A - x11_4A - x11_5A - x11_6A - x11_7A - x11_8A - x11_10A - x11_13A = 0 ;

vertice12: x11_12L + x7_12L + x9_12L + x0_12A + x1_12A + x3_12A + x5_12A + x6_12A + x8_12A + x13_12A
- x12_11L - x12_7L - x12_9L - x12_0A - x12_1A - x12_3A - x12_5A - x12_6A - x12_8A - x12_13A = 0 ;

vertice13: x5_13L + x6_13L + x8_13L + x0_13A + x1_13A + x2_13A + x3_13A + x4_13A + x7_13A + x9_13A + x10_13A + x11_13A + x12_13A
- x13_5L - x13_6L - x13_8L - x13_0A - x13_1A - x13_2A - x13_3A - x13_4A - x13_7A - x13_9A - x13_10A - x13_11A - x13_12A = 0 ;

```

Figura 11 - 1ª Restrição do Modelo Anterior

Figura 12 - 2ª Restrição do Modelo Anterior

Tal como é possível observar estes dois conjuntos de restrições incluem inúmeras hipóteses e variáveis fazendo com que este modelo esteja mais propício a erros, diminuindo desta forma a sua eficácia e aumentando a sua complexidade. Na 1ª Restrição nós consideramos que as arestas têm de ser percorridas pelo menos uma vez. Isto é algo repetitivo, uma vez que o objetivo do problema é que todas essas arestas sejam percorridas. O que nós necessitamos de saber é quais as arestas que devem ser repetidas e quais os percursos aéreos a tomar de forma a que o drone consiga percorrer todas as linhas de alta tensão. É importante salientar que também é desnecessário considerarmos o sentido que cada aresta poderá ser percorrida, uma vez que não se trata de um grafo orientado.

Finalmente, o output dado pelo *LPSolve* foi o seguinte:

Source Matrix Options Result									
Objective Constraints Sensitivity									
Variables	MILP ...	MILP ...	MILP ...	MILP ...	MILP ...	MILP ...	MILP ...	r...	
	110,11	109,31	107,29	106,78	103,54	101,29	95,41	95,41	
x01L	1	1	1	1	1	1	1	1	
x40L	1	1	1	1	1	1	1	1	
x12L	1	1	1	1	1	1	1	1	
x16L	1	1	1	1	1	1	1	1	
x61L	0	0	0	0	1	1	1	1	
x23L	1	1	1	1	1	1	1	1	
x52A	1	1	1	1	1	1	1	1	
x25L	1	1	1	1	1	1	1	1	
x35L	1	1	1	1	1	1	1	1	
x34L	1	1	1	1	1	1	1	1	
x43L	0	0	0	0	0	0	1	1	
x74L	1	0	1	0	0	1	1	1	
x5_13L	1	1	1	1	1	1	1	1	
x13_6L	1	0	1	0	0	1	1	1	
x69L	1	1	1	1	0	1	1	1	
x12_7L	1	0	1	0	0	1	1	1	
x8_13L	1	1	1	1	1	1	1	1	
x13_8L	0	0	0	0	1	0	1	1	
x10_8L	0	0	1	1	1	1	1	1	
x8_11L	1	1	1	1	1	1	1	1	
x9_10L	1	1	2	1	1	1	1	1	
x10_9L	1	0	0	0	1	1	1	1	
x9_12L	1	0	0	0	0	1	1	1	
x11_10L	0	0	0	1	1	1	1	1	
x11_12L	0	1	1	1	1	0	1	1	
x12_11L	1	0	0	1	1	1	1	1	
x10L	0	0	0	0	0	0	0	0	
x02A	0	0	0	0	0	0	0	0	
x20A	0	0	0	0	0	0	0	0	
x03A	0	0	0	0	0	0	0	0	
x30A	0	0	0	0	0	0	0	0	
x04A	0	0	0	0	0	0	0	0	
x40A	0	0	0	0	0	0	0	0	
x04L	0	0	0	0	0	0	0	0	
x05A	0	0	0	0	0	0	0	0	

Figura 13 - Resultado do Modelo Anterior

Observando os dados resultantes verificamos que a última coluna foi obtida através de múltiplas iterações até alcançar a solução ótima. Esta coluna encontra-se organizada de forma decrescente, logo, podemos concluir que em todas as variáveis com resultado diferente de 0, as respectivas arestas foram percorridas pelo menos uma vez. Como as únicas variáveis que apresentam uma solução são iguais a 1, significa que essas arestas apenas foram percorridas uma vez, já que a variável X_{ijk} representa o nº de vezes que uma aresta é percorrida.

Analisando agora a solução ótima vemos que as arestas percorridas são as mesmas que o grupo obteve com a S.O do modelo apresentado neste projeto, tal como a distância total percorrida. É importante referir que este fator reforça novamente a validação do nosso modelo.

Comparando ambos os modelos, vemos que o modelo mais recente, e proposto pelo grupo para a resolução do problema, é aquele que melhor traduz as regras de

funcionamento do sistema, de uma forma simples, clara e objetiva. Nessa resolução apenas consideramos as arestas que queremos repetir ou os percursos aéreos a tomar, excluindo assim as linhas de alta tensão (arestas do grafo) que o drone já tem de percorrer obrigatoriamente uma vez. Sendo que, o fator decisivo neste problema é como o drone irá reposicionar-se e tomar certos caminhos de forma a percorrer todas as linhas, considerando sempre o menor custo possível na distância percorrida. Desta forma, ao contrário do modelo antigo, o custo que nós queremos descobrir é o custo da distância das “arestas extras” que têm de ser percorridas de forma a conseguir passar por todas as linhas, tendo em conta que o custo das arestas já definidas é de 81 unidades de distância.

Assim sendo, nós acabamos por considerar menos variáveis do que no modelo antigo. Já que nesse, além dos percursos aéreos e das linhas de alta tensão, incluímos também o sentido em que cada aresta é percorrida, prejudicando assim a eficiência na sua resolução, uma vez que, essa abordagem aplica-se melhor aos grafos orientados.

Conclusão

Concluimos então, no final de analisarmos as duas resoluções, que realmente aquela que foi apresentada e elaborada é a resolução mais adequada, pois é aquela que inclui menos variáveis e traduz a medida desejada de eficiência, tendo como base a Teoria dos Grafos. Na nossa opinião, ambas as resoluções foram importantes, de forma a permitir-nos ter um sentido crítico do nosso trabalho, a encontrarmos espaço para melhoria e, consequentemente, a um maior aproveitamento da unidade curricular.