

Universidade do Minho  
Escola de Engenharia

# Investigação Operacional DRONE

## Trabalho Prático 1

**2021/2022**

**Autores:**

a91671 João Manuel Novais da Silva

a91697 Luís Filipe Fernandes Vilas

a91660 Pedro António Pires Correia Leite Sequeira

**Docente:**

José Manuel Vasconcelos Valério Carvalho

Braga

24 de março de 2022

# Índice

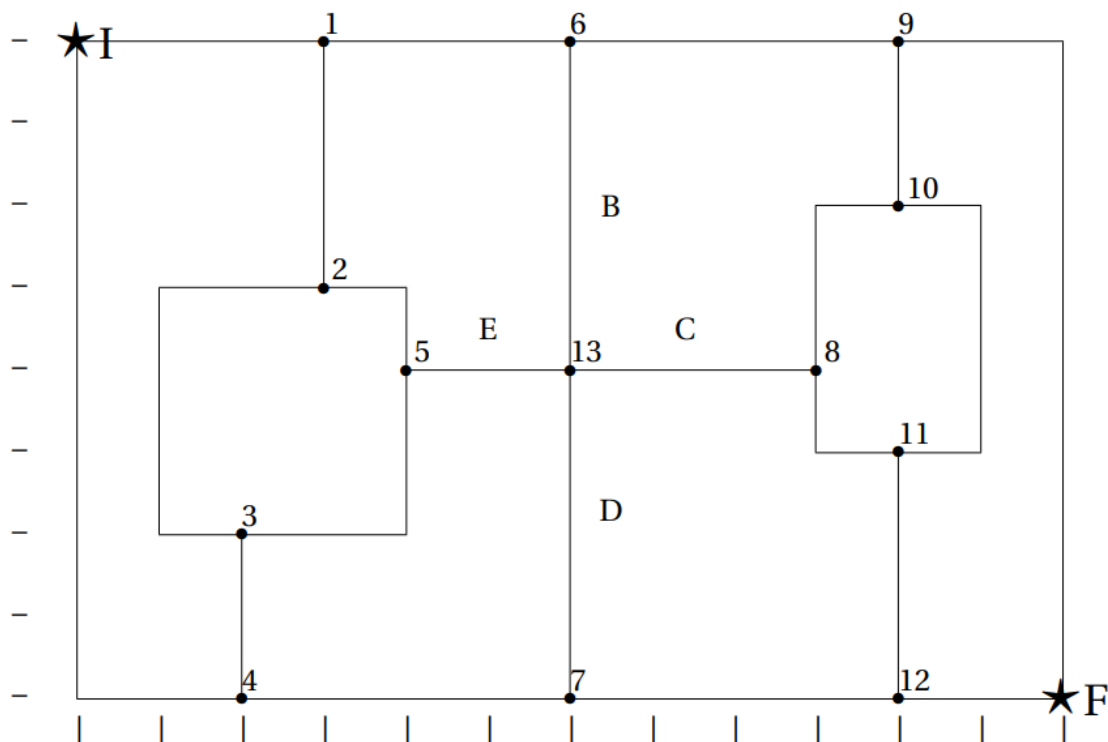
<b>1.</b>	<b>Introdução.....</b>	<b>1</b>
1.1.	Objetivos .....	2
<b>2.</b>	<b>Problema.....</b>	<b>2</b>
<b>3.</b>	<b>Formulação do Problema .....</b>	<b>3</b>
3.1.	Variáveis de Decisão .....	4
3.2.	Função Objetivo.....	4
3.3.	Restrições .....	5
<b>4.</b>	<b>Modelação: LP Solve .....</b>	<b>6</b>
4.1.	Ficheiro de Input.....	6
4.2.	Ficheiros de Output .....	7
<b>5.</b>	<b>Interpretação do Resultado: Solução Ótima.....</b>	<b>8</b>
<b>6.</b>	<b>Validação do Modelo .....</b>	<b>10</b>
<b>7.</b>	<b>Conclusão.....</b>	<b>13</b>

# 1. Introdução

Neste trabalho aborda-se o problema do “Drone”, sendo este bastante semelhante ao problema do Carteiro Chinês. O cenário apresentado consiste em determinar o percurso em que todas as arestas de um grafo são percorridas, pelo menos uma vez, minimizando a distância total percorrida. As arestas podem ser percorridas em qualquer sentido, e pode ser necessário atravessar a mesma aresta mais de uma vez.

Este caso em particular, ocorre quando um veículo não tripulado tem de inspecionar linhas de transporte de energia elétrica em alta tensão para verificar se há vegetação a interferir com as linhas. Para reposicionar o drone para recomençar a inspeção de uma nova linha não é necessário seguir as linhas de alta tensão, bastando fazer o percurso mais curto através do ar. Na figura abaixo, encontra-se o mapa de linhas de alta tensão exposto no problema.

O objetivo deste trabalho prático é resolver o problema supramencionado utilizando programação linear.



*Figura 1 - Mapa de Linhas de Alta Tensão*

## 1.1. Objetivos

Este trabalho consiste em encontrar a solução ótima para o problema. Para tal tem de se encontrar um caminho que comece em I (ponto inicial) e termine em F (ponto final) 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) pelo menos uma vez. Trata-se de um percurso denominado por caminho Euleriano.

É importante referir também que cada aresta pode ser percorrida mais de uma vez, e em qualquer sentido.

## 2. Problema

De forma a diferenciar as soluções dos diferentes grupos, foi implementado o método pedido para remoção de arestas do grafo.

O maior número de inscrição entre os elementos do grupo é 91697, pelo que se removeu a aresta C, o que resultou no seguinte grafo:

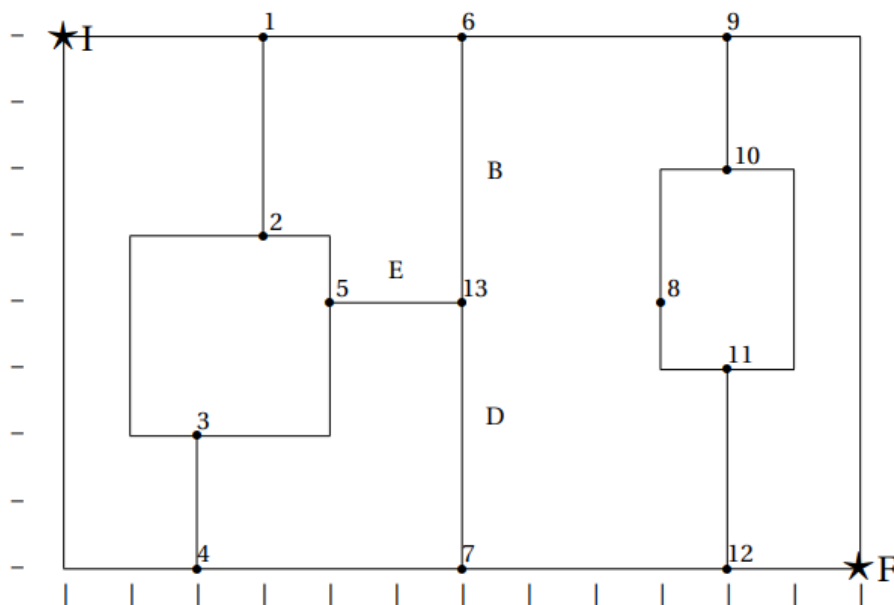


Figura 2 - Grafo usado na resolução do Problema

### 3. Formulação do Problema

Tal como dito anteriormente, o problema consiste no cálculo do caminho mais curto de um drone que inspeciona linhas de transporte de energia elétrica em alta tensão.

Pretende-se que o drone verifique todas as linhas (arestas), tendo como dados as distâncias entre cada um dos vértices. De forma a resolver isto, recorreu-se a emparelhamentos. Um 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 caminho Euleriano. O teorema de Euler para grafos define este conceito da seguinte forma: "Um grafo  $G$  conexo possui caminho Euleriano se e somente se ele tem exatamente zero ou dois vértices de grau ímpar".

Este tipo de grafo, permite que, partindo de um vértice qualquer, possamos percorrer todas as arestas passando por elas apenas uma única vez. Isto no problema descrito seria uma solução ideal, no entanto não é aplicável pois como foi dito é necessário que o grafo tenha todos os vértices de grau par (exceto, neste caso, o inicial e final), e o apresentado anteriormente tem 12 vértices de grau ímpar. É possível perceber que será obrigatório repetir algumas arestas, sendo que a dúvida agora recai em saber que arestas repetir (ou arestas aéreas a percorrer), tendo em conta que o objetivo é minimizar o custo. Esta duplicação é feita de modo que o grafo resultante tenha os vértices acima mencionados de grau par.

Variáveis de decisão, função objetivo e restrições são três termos que serão utilizados neste trabalho. A solução de um problema de Programação Linear contém sempre estes três elementos.

Todos os problemas de Programação Linear pretendem obter uma solução ótima para uma dada situação. Esta solução é resultante de um conjunto de decisões - variáveis de decisão - que são todas as opções que temos num problema.

A otimização de um problema de Programação Linear objetivará a maximização ou minimização de um determinado objetivo. Desse modo, a Função Objetivo é uma expressão matemática que quantifica a solução de um dado problema. No entanto, haverá sempre algo que limita a otimização do mesmo. Assim sendo, as restrições são

limitantes aos valores das variáveis de decisão. As restrições são expressas em equações e/ou inequações matemáticas que traduzem um limitante físico à solução do problema.

### 3.1. Variáveis de Decisão

As variáveis de decisão (binárias, com o objetivo de saber se estão no emparelhamento ótimo) representam as arestas adicionais que irão ser percorridas pelo ar. Por isso, é necessário que exista uma variável de decisão por cada ligação possível entre vértices. Deste modo, as variáveis têm a forma  $x_{ij}$ , onde  $i$  e  $j$  representam as extremidades da aresta representada. Além disto não é permitido ter  $i = j$  pois isso não representaria nenhuma aresta e como não se consideram pares repetidos, temos em conta  $j > i$  de forma a evitar que sejam selecionados em simultâneo, pares como  $x_{12}$  e  $x_{21}$  (ambas as formas representam o mesmo par):

$$x_{ij} \in \{0, 1\}, \forall i, j \in \{0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14\}, i \neq j \wedge j > i$$

É de notar que os vértices 0 e 14 surgiram da necessidade de identificar os vértices inicial e final, respetivamente.

Na realidade, cada variável de decisão traduz uma possibilidade de aresta a adicionar ao grafo de forma a tornar esses mesmos vértices de grau par. Além disto, a escolha das arestas depende do custo, logo os coeficientes de cada uma das variáveis corresponde à menor distância entre os vértices que formam o par.

Note-se que as arestas percorridas sobre as estradas não são contabilizadas visto que, no contexto deste problema, estas serão sempre percorridas independentemente da solução admissível, permanecendo sempre como constantes no modelo.

### 3.2. Função Objetivo

Visto que queremos encontrar o emparelhamento ótimo, a função objetivo será obrigatoriamente de minimização (minimiza a distância percorrida nos vértices a escolher) e, dará como resultado o custo total mínimo dos caminhos fora das linhas a percorrer pelo drone.

A função objetivo é do tipo  $\min z = c_1x_1 + c_2x_2 + \dots + c_nx_n$  onde os coeficientes  $c_1, c_2, \dots, c_n$  são os coeficientes de distâncias (que são dados conhecidos) e  $x_1, x_2, \dots, x_n$ , representam as variáveis de decisão que serão determinadas. Neste caso, tal como mencionado acima, representam os pares de vértices que constituem uma possível aresta, sem colocar em causa a paridade dos vértices já pares.

Por exemplo, neste caso, a aresta  $x_{18}$  não é considerada pois, caso fosse feita esta ligação, o vértice 8, passaria a ser de grau ímpar, o que é exatamente o contrário daquilo que pretendemos.

$$\min z = \sum_{i=0}^6 \sum_{j=i+1}^7 c_{ij} * x_{ij} + \sum_{i=0}^7 \sum_{j=9}^{14} c_{ij} * x_{ij} + \sum_{i=9}^{13} \sum_{j=i+1}^{14} c_{ij} * x_{ij}$$

### 3.3. Restrições

As restrições garantem que o emparelhamento é formado apenas por pares válidos, ou seja, como foi explicado anteriormente, tem de se garantir que o vértice V (de grau ímpar) está incluído num e num só par desse mesmo emparelhamento ótimo. Caso contrário, verificar-se-ia uma de duas situações: ou o vértice se tornava par, mas iriam ser adicionadas arestas desnecessárias (que representava um aumento da distância percorrida), ou então o vértice continuava a ser de grau ímpar, o que não resolvia 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, do tipo:

$$\sum x_{ji} + \sum x_{ij} = 1,$$

onde no primeiro somatório j percorre todos os vértices de grau ímpar menores que i, e no segundo somatório j percorre todos os vértices de grau ímpar maiores que i.

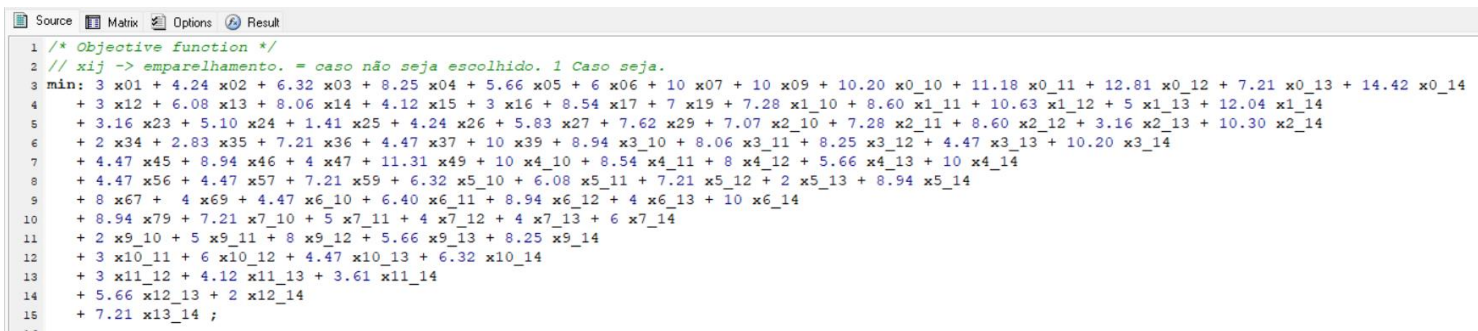
Por exemplo a restrição do vértice  $i = 3$  seria:

$$[x_{03} + x_{13} + x_{23}] + [x_{34} + x_{35} + x_{36} + x_{37} + x_{39} + x_{310} + x_{311} + x_{312} + x_{313} + x_{314}] = 1$$

## 4. Modelação: LP Solve

Para a resolução do problema enunciado anteriormente recorreu-se à utilização do software de Programação Linear, recomendado pelo docente, LP Solve.

### 4.1. Ficheiro de Input

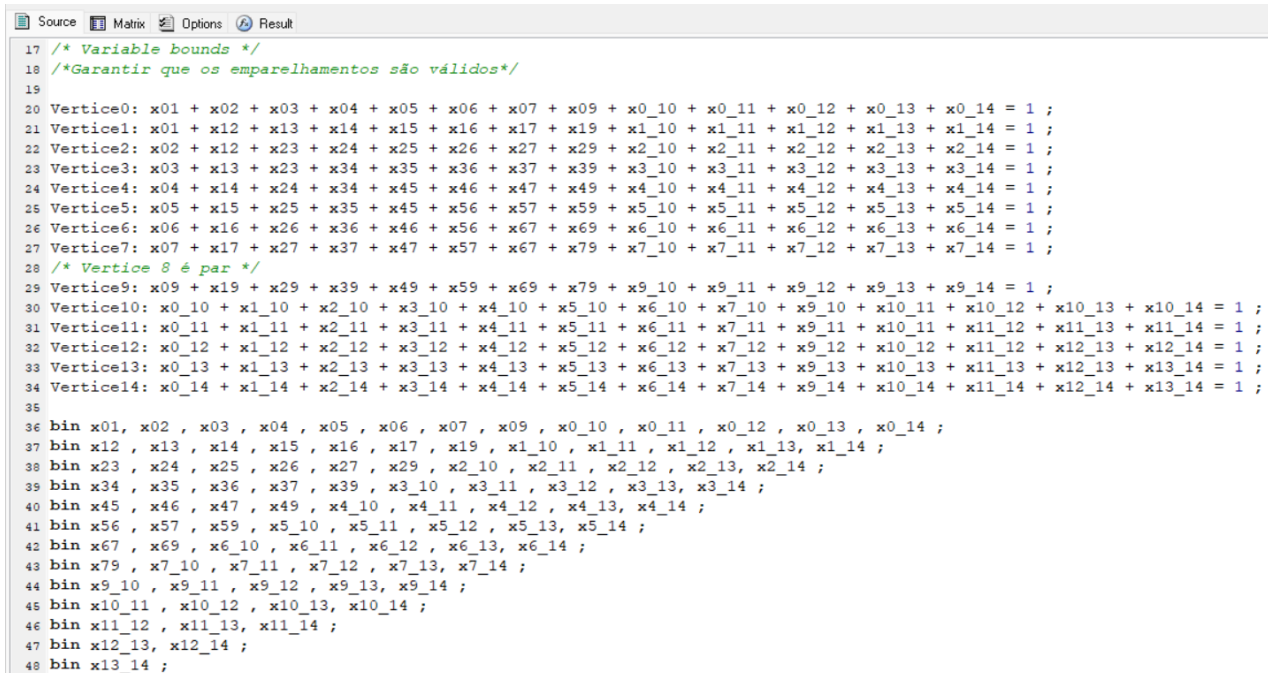


```

1 /* Objective function */
2 // xij -> emparelhamento. = caso não seja escolhido. 1 Caso seja.
3 min: 3 x01 + 4.24 x02 + 6.32 x03 + 8.25 x04 + 5.66 x05 + 6 x06 + 10 x07 + 10 x09 + 10.20 x0_10 + 11.18 x0_11 + 12.81 x0_12 + 7.21 x0_13 + 14.42 x0_14
4 + 3 x12 + 6.08 x13 + 8.06 x14 + 4.12 x15 + 3 x16 + 8.54 x17 + 7 x19 + 7.28 x1_10 + 8.60 x1_11 + 10.63 x1_12 + 5 x1_13 + 12.04 x1_14
5 + 3.16 x23 + 5.10 x24 + 1.41 x25 + 4.24 x26 + 5.83 x27 + 7.62 x29 + 7.07 x2_10 + 7.28 x2_11 + 8.60 x2_12 + 3.16 x2_13 + 10.30 x2_14
6 + 2 x34 + 2.83 x35 + 7.21 x36 + 4.47 x37 + 10 x39 + 8.94 x3_10 + 8.06 x3_11 + 8.25 x3_12 + 4.47 x3_13 + 10.20 x3_14
7 + 4.47 x45 + 8.94 x46 + 4 x47 + 11.31 x49 + 10 x4_10 + 8.54 x4_11 + 8 x4_12 + 5.66 x4_13 + 10 x4_14
8 + 4.47 x56 + 4.47 x57 + 7.21 x59 + 6.32 x5_10 + 6.08 x5_11 + 7.21 x5_12 + 2 x5_13 + 8.94 x5_14
9 + 8 x67 + 4 x69 + 4.47 x6_10 + 6.40 x6_11 + 8.94 x6_12 + 4 x6_13 + 10 x6_14
10 + 8.94 x79 + 7.21 x7_10 + 5 x7_11 + 4 x7_12 + 4 x7_13 + 6 x7_14
11 + 2 x9_10 + 5 x9_11 + 8 x9_12 + 5.66 x9_13 + 8.25 x9_14
12 + 3 x10_11 + 6 x10_12 + 4.47 x10_13 + 6.32 x10_14
13 + 3 x11_12 + 4.12 x11_13 + 3.61 x11_14
14 + 5.66 x12_13 + 2 x12_14
15 + 7.21 x13_14 ;

```

Figura 3 - Função Objetivo



```

17 /* Variable bounds */
18 /*Garantir que os emparelhamentos são válidos*/
19
20 Vertice0: x01 + x02 + x03 + x04 + x05 + x06 + x07 + x09 + x0_10 + x0_11 + x0_12 + x0_13 + x0_14 = 1 ;
21 Vertice1: x01 + x12 + x13 + x14 + x15 + x16 + x17 + x19 + x1_10 + x1_11 + x1_12 + x1_13 + x1_14 = 1 ;
22 Vertice2: x02 + x12 + x23 + x24 + x25 + x26 + x27 + x29 + x2_10 + x2_11 + x2_12 + x2_13 + x2_14 = 1 ;
23 Vertice3: x03 + x13 + x23 + x34 + x35 + x36 + x37 + x39 + x3_10 + x3_11 + x3_12 + x3_13 + x3_14 = 1 ;
24 Vertice4: x04 + x14 + x24 + x34 + x45 + x46 + x47 + x49 + x4_10 + x4_11 + x4_12 + x4_13 + x4_14 = 1 ;
25 Vertice5: x05 + x15 + x25 + x35 + x45 + x56 + x57 + x59 + x5_10 + x5_11 + x5_12 + x5_13 + x5_14 = 1 ;
26 Vertice6: x06 + x16 + x26 + x36 + x46 + x56 + x67 + x69 + x6_10 + x6_11 + x6_12 + x6_13 + x6_14 = 1 ;
27 Vertice7: x07 + x17 + x27 + x37 + x47 + x57 + x67 + x79 + x7_10 + x7_11 + x7_12 + x7_13 + x7_14 = 1 ;
28 /* Vertice 8 é par */
29 Vertice9: x09 + x19 + x29 + x39 + x49 + x59 + x69 + x79 + x9_10 + x9_11 + x9_12 + x9_13 + x9_14 = 1 ;
30 Vertice10: x0_10 + x1_10 + x2_10 + x3_10 + x4_10 + x5_10 + x6_10 + x7_10 + x9_10 + x10_11 + x10_12 + x10_13 + x10_14 = 1 ;
31 Vertice11: x0_11 + x1_11 + x2_11 + x3_11 + x4_11 + x5_11 + x6_11 + x7_11 + x9_11 + x10_11 + x11_12 + x11_13 + x11_14 = 1 ;
32 Vertice12: x0_12 + x1_12 + x2_12 + x3_12 + x4_12 + x5_12 + x6_12 + x7_12 + x9_12 + x10_12 + x11_12 + x12_13 + x12_14 = 1 ;
33 Vertice13: x0_13 + x1_13 + x2_13 + x3_13 + x4_13 + x5_13 + x6_13 + x7_13 + x9_13 + x10_13 + x11_13 + x12_13 + x13_14 = 1 ;
34 Vertice14: x0_14 + x1_14 + x2_14 + x3_14 + x4_14 + x5_14 + x6_14 + x7_14 + x9_14 + x10_14 + x11_14 + x12_14 + x13_14 = 1 ;
35
36 bin x01, x02, x03, x04, x05, x06, x07, x09, x0_10, x0_11, x0_12, x0_13, x0_14 ;
37 bin x12, x13, x14, x15, x16, x17, x19, x1_10, x1_11, x1_12, x1_13, x1_14 ;
38 bin x23, x24, x25, x26, x27, x29, x2_10, x2_11, x2_12, x2_13, x2_14 ;
39 bin x34, x35, x36, x37, x39, x3_10, x3_11, x3_12, x3_13, x3_14 ;
40 bin x45, x46, x47, x49, x4_10, x4_11, x4_12, x4_13, x4_14 ;
41 bin x56, x57, x59, x5_10, x5_11, x5_12, x5_13, x5_14 ;
42 bin x67, x69, x6_10, x6_11, x6_12, x6_13, x6_14 ;
43 bin x79, x7_10, x7_11, x7_12, x7_13, x7_14 ;
44 bin x9_10, x9_11, x9_12, x9_13, x9_14 ;
45 bin x10_11, x10_12, x10_13, x10_14 ;
46 bin x11_12, x11_13, x11_14 ;
47 bin x12_13, x12_14 ;
48 bin x13_14 ;

```

Figura 4 - Restrições



## 4.2. Ficheiros de Output

### 4.2.1. Log

```

Log  Messages
Model name: 'LPSolver' - run #1
Objective: Minimize(R0)

SUBMITTED
Model size:      14 constraints,      91 variables,      182 non-zeros.
Sets:           0 GUB,              0 SOS.

Using DUAL simplex for phase 1 and PRIMAL simplex for phase 2.
The primal and dual simplex pricing strategy set to 'Devex'.

Relaxed solution      19.41 after      15 iter is B&B base.

Feasible solution      19.41 after      15 iter,      0 nodes (gap 0.0%)

Optimal solution      19.41 after      15 iter,      0 nodes (gap 0.0%).
Relative numeric accuracy ||*|| = 2.22045e-016

MEMO: lp_solve version 5.5.2.11 for 32 bit OS, with 64 bit REAL variables.
In the total iteration count 15, 0 (0.0%) were bound flips.
There were 0 refactorizations, 0 triggered by time and 0 by density.
... on average 15.0 major pivots per refactorization.
The largest [LUSOL v2.2.1.0] fact(B) had 15 NZ entries, 1.0x largest basis.
The maximum B&B level was 1, 0.0x MIP order, 1 at the optimal solution.
The constraint matrix inf-norm is 1, with a dynamic range of 1.
Time to load data was 0.007 seconds, presolve used 0.005 seconds,
... 0.016 seconds in simplex solver, in total 0.028 seconds.

```

Figura 5 - Ficheiro de Log

### 4.2.2. Resultados

Source	Matrix	Options	Result
Objective	Constraints	Sensitivity	
Variables	MILP ...	result	
x01	1	1	
x02	0	0	
x03	0	0	
x04	0	0	
x05	0	0	
x06	0	0	
x07	0	0	
x09	0	0	
x0_10	0	0	
x0_11	0	0	
x0_12	0	0	
x0_13	0	0	
x0_14	0	0	
x12	0	0	
x13	0	0	
x14	0	0	
x15	0	0	
x16	0	0	
x17	0	0	
x19	0	0	
x1_10	0	0	
x1_11	0	0	
x1_12	0	0	
x1_13	0	0	
x1_14	0	0	
x23	0	0	
x24	0	0	
x25	1	1	
x26	0	0	
x27	0	0	
x29	0	0	
x2_10	0	0	
x2_11	0	0	
x2_12	0	0	
x2_13	0	0	

Source	Matrix	Options	Result
Objective	Constraints	Sensitivity	
Variables	MILP ...	result	
x2_14	0	0	
x34	1	1	
x35	0	0	
x36	0	0	
x37	0	0	
x39	0	0	
x3_10	0	0	
x3_11	0	0	
x3_12	0	0	
x3_13	0	0	
x3_14	0	0	
x45	0	0	
x46	0	0	
x47	0	0	
x49	0	0	
x4_10	0	0	
x4_11	0	0	
x4_12	0	0	
x4_13	0	0	
x4_14	0	0	
x56	0	0	
x57	0	0	
x59	0	0	
x5_10	0	0	
x5_11	0	0	
x5_12	0	0	
x5_13	0	0	
x5_14	0	0	
x67	0	0	
x69	1	1	
x6_10	0	0	
x6_11	0	0	
x6_12	0	0	
x6_13	0	0	
x6_14	0	0	

Source	Matrix	Options	Result
Objective	Constraints	Sensitivity	
Variables	MILP ...	result	
x79	0	0	
x7_10	0	0	
x7_11	0	0	
x7_12	0	0	
x7_13	1	1	
x7_14	0	0	
x9_10	0	0	
x9_11	0	0	
x9_12	0	0	
x9_13	0	0	
x9_14	0	0	
x10_11	1	1	
x10_12	0	0	
x10_13	0	0	
x10_14	0	0	
x11_12	0	0	
x11_13	0	0	
x11_14	0	0	
x12_13	0	0	
x12_14	1	1	
x13_14	0	0	

Figura 6 - Output do LP Solve

Com base nos resultados da figura 6, pode-se verificar que a solução ótima corresponde aos pares:  $\{(0,1), (2,5), (3,4), (6,9), (7,13), (10,11), (12,14)\}$ .

Além disso, verifica-se também que a distância total percorrida fora do trajeto das linhas de alta tensão é de 19,41 (unidades de distância). Este valor, representa a soma das distâncias das arestas cuja variável tem valor 1.

## 5. Interpretação do Resultado: Solução Ótima

Com os pares acima enumerados é possível encontrar o percurso ótimo no grafo de modo que todos os seus vértices (à exceção do inicial e final) sejam de grau par, ou seja, obtendo-se um grafo Euleriano (figura 7). Assim sendo, sabemos que é possível percorrer todas as arestas e passar pelas mesmas apenas uma só vez.

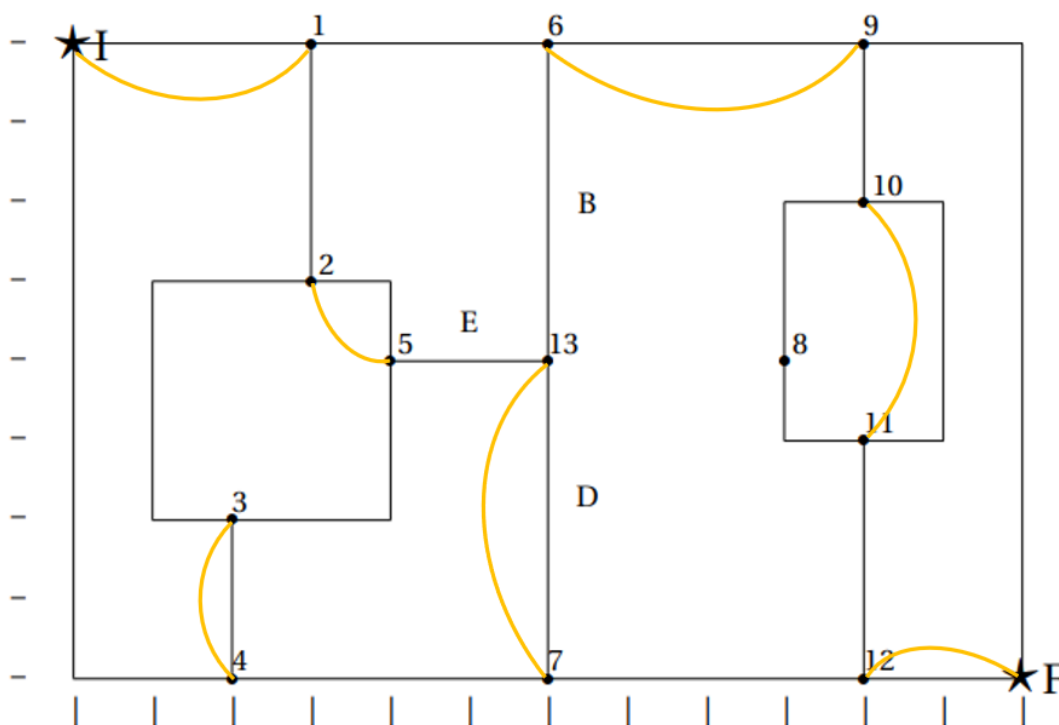


Figura 7 - Grafo incluindo a solução ótima

Existem várias possibilidades de percursos a realizar, sabendo agora quais as linhas de alta tensão que iremos repetir, as arestas diagonais (pelo ar) a percorrer, e as linhas de alta tensão por onde o drone apenas passa uma vez, obtemos o seguinte exemplo de percurso, representado na figura 8.

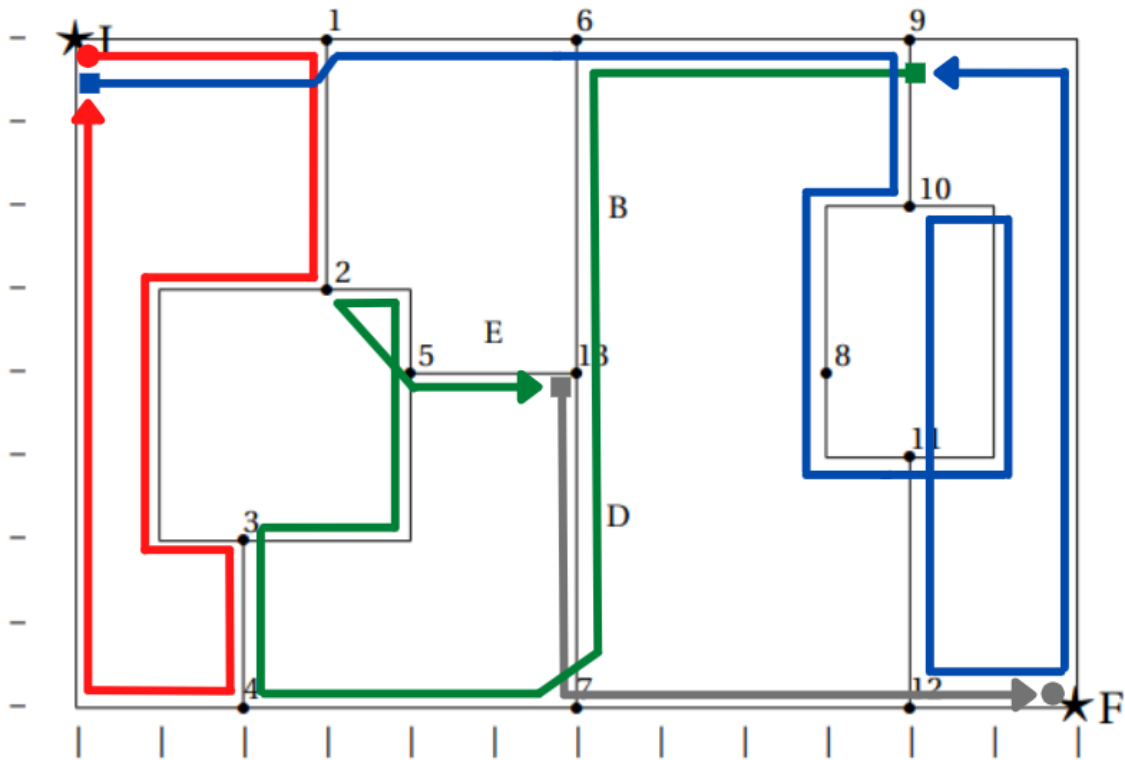


Figura 8 - Possibilidade de percurso ótimo

O exemplo ilustrado acima é constituído pelo seguinte percurso:

**Vermelho** → **Azul** → **Verde** → **Cinzento**

Ou seja:

**I → 1 → 2 → 3 → 4 → I → 1 → 6 → 9 → 10 → 8 → 11 → 10 → 11 → 12**  
**→ F → 9 → 6 → 13 → 7 → 4 → 3 → 5 → 2 → 5 → 13 → 7 → 12 → F**

Considerando o grafo da figura 2 e somando-se a distância de cada aresta percorrendo a totalidade dos fios de alta tensão (valores considerados constantes na resolução do problema, visto que se sabia desde o início que era obrigatória a passagem de pelo menos uma vez nos mesmos), obteve-se:

$$8 + 3 + 3 + 4 + 2 + 8 + 2 + 4 + 4 + 2 + 3 + 2 + 3 + 1 + 2 + 2 + 1 + 1 + 2 + 2 + 4 + 4 + 2 + 1 + 2 + 1 + 1 + 3 + 1 + 3 + 1 = \mathbf{82 \text{ UD}}$$

Sendo que a soma obtida pelas arestas que terão de se repetir (resultado obtido no LP Solver) é 19,41 UD, a distância total mínima percorrida será de:

$$\mathbf{82 \text{ UD} + 19.41 \text{ UD} = 101,41 \text{ UD}}$$

## 6. Validação do Modelo

Após obter o resultado ótimo no LP Solve foi necessário verificar que a solução obtida satisfaz todas as condições idealizadas inicialmente e se adequa ao sistema na realidade.

Assim, para validar o modelo foram efetuadas as seguintes verificações:

- Nenhuma distância entre vértices é negativa ou nula (figura 9);

$$\text{distancia total percorrida} > 0 \rightarrow 101.41 \text{ UD} > 0$$

		x	0	3	3	2	2	4	6	6	9	10	10	10	10	6	12
		y	8	8	5	2	0	4	8	0	4	8	6	3	0	4	0
x	y		I	1	2	3	4	5	6	7	8	9	10	11	12	13	F
0	8	I	0,00	3,00	4,24	6,32	8,25	5,66	6,00	10,00	9,85	10,00	10,20	11,18	12,81	7,21	14,42
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	12,04
3	5	2			0,00	3,16	5,10	1,41	4,24	5,83	6,08	7,62	7,07	7,28	8,60	3,16	10,30
2	2	3				0,00	2,00	2,83	7,21	4,47	7,28	10,00	8,94	8,06	8,25	4,47	10,20
2	0	4					0,00	4,47	8,94	4,00	8,06	11,31	10,00	8,54	8,00	5,66	10,00
4	4	5						0,00	4,47	4,47	5,00	7,21	6,32	6,08	7,21	2,00	8,94
6	8	6							0,00	8,00	5,00	4,00	4,47	6,40	8,94	4,00	10,00
6	0	7								0,00	5,00	8,94	7,21	5,00	4,00	4,00	6,00
9	4	8									0,00	4,12	2,24	1,41	4,12	3,00	5,00
10	8	9										0,00	2,00	5,00	8,00	5,66	8,25
10	6	10											0,00	3,00	6,00	4,47	6,32
10	3	11												0,00	3,00	4,12	3,61
10	0	12													0,00	5,66	2,00
6	4	13														0,00	7,21
12	0	F															0,00

Figura 9 - Distâncias euclidianas entre vértices

- Os emparelhamentos escolhidos não têm vértices repetidos, ou seja, para cada vértice foi escolhido apenas um arco (figura 10);

Source Matrix Options Result		
Objective Constraints Sensitivity		
Constraints	MILP ...	result
	19,41	19,41
Vertice0	1	1
Vertice1	1	1
Vertice2	1	1
Vertice3	1	1
Vertice4	1	1
Vertice5	1	1
Vertice6	1	1
Vertice7	1	1
Vertice9	1	1
Vertice10	1	1
Vertice11	1	1
Vertice12	1	1
Vertice13	1	1
Vertice14	1	1

Figura 10 - Valores das Restrições

- Todos os vértices do grafo inicialmente ímpares ficam com grau par – caminho Euleriano (figura 7);
- É possível efetuar um percurso (com origem em I e destino no vértice F) que passa em todas as arestas necessárias apenas uma vez, acrescentado unicamente os emparelhamentos selecionados (solução ótima – figura 8);
- Todas as variáveis são binárias;
- O LP Solve fornece um resultado válido, pois substituindo, manualmente, os valores obtidos nas restrições e na função objetivo, a solução coincide com a solução ótima obtida:

- Função Objetivo:

```
min: 3 x01 + 4.24 x02 + 6.32 x03 + 8.25 x04 + 5.66 x05 + 6 x06 + 10 x07 + 10 x09 + 10.20 x0_10 + 11.18 x0_11 + 12.81 x0_12 + 7.21 x0_13 + 14.42 x0_14
+ 3 x12 + 6.08 x13 + 8.06 x14 + 4.12 x15 + 3 x16 + 8.54 x17 + 7 x19 + 7.28 x1_10 + 8.60 x1_11 + 10.63 x1_12 + 5 x1_13 + 12.04 x1_14
+ 3.16 x23 + 5.10 x24 + 1.41 x25 + 4.24 x26 + 5.83 x27 + 7.62 x29 + 7.07 x2_10 + 7.28 x2_11 + 8.60 x2_12 + 3.16 x2_13 + 10.30 x2_14
+ 2 x34 + 2.83 x35 + 7.21 x36 + 4.47 x37 + 10 x39 + 8.94 x3_10 + 8.06 x3_11 + 8.25 x3_12 + 4.47 x3_13 + 10.20 x3_14
+ 4.47 x45 + 8.94 x46 + 4 x47 + 11.31 x49 + 10 x4_10 + 8.54 x4_11 + 8 x4_12 + 5.66 x4_13 + 10 x4_14
+ 4.47 x56 + 4.47 x57 + 7.21 x59 + 6.32 x5_10 + 6.08 x5_11 + 7.21 x5_12 + 2 x5_13 + 8.94 x5_14
+ 8 x67 + 4 x69 + 4.47 x6_10 + 6.40 x6_11 + 8.94 x6_12 + 4 x6_13 + 10 x6_14
+ 8.94 x79 + 7.21 x7_10 + 5 x7_11 + 4 x7_12 + 4 x7_13 + 6 x7_14
+ 2 x9_10 + 5 x9_11 + 8 x9_12 + 5.66 x9_13 + 8.25 x9_14
+ 3 x10_11 + 6 x10_12 + 4.47 x10_13 + 6.32 x10_14
+ 3 x11_12 + 4.12 x11_13 + 3.61 x11_14
+ 5.66 x12_13 + 2 x12_14
+ 7.21 x13_14 ;
```

Com os resultados do LP Solve:

$$3 * 1 + 4.24 * 0 + 6.32 * 0 + 8.25 * 0 + 5.66 * 0 + 6 * 0 + 10 * 0 + 10 * 0 + 10.20 * 0 + 11.18 * 0 + 12.81 * 0 + 7.21 * 0 + 14.42 * 0 + 3 * 0 + 6.08 * 0 + 8.06 * 0 + 4.12 * 0 + 3 * 0 + 8.54 * 0 + 7 * 0 + 7.28 * 0 + 8.60 * 0 + 10.63 * 0 + 5 * 0 + 12.04 * 0 + 3.16 * 0 + 5.10 * 0 + 1.41 * 1 + 4.24 * 0 + 5.83 * 0 + 7.62 * 0 + 7.07 * 0 + 7.28 * 0 + 8.6 * 0 + 3.16 * 0 + 10.30 * 0 + 2 * 1 + 2.83 * 0 + 7.21 * 0 + 4.47 * 0 + 10 * 0 + 8.94 * 0 + 8.06 * 0 + 8.25 * 0 + 4.47 * 0 + 10.20 * 0 + 4.47 * 0 + 8.94 * 0 + 4 * 0 + 11.31 * 0 + 10 * 0 + 8.54 * 0 + 8 * 0 + 5.66 * 0 + 10 * 0 + 4.47 * 0 + 4.47 * 0 + 7.21 * 0 + 6.32 * 0 + 6.08 * 0 + 7.21 * 0 + 2 * 0 + 8.94 * 0 + 8 * 0 + 4 * 1 + 4.47 * 0 + 7.4 * 0 + 8.94 * 0 + 4 * 0 + 10 * 0 + 2 * 0 + 8.94 * 0 + 7.21 * 0 + 5 * 0 + 4 * 0 + 4 * 1 + 6 * 0 + 2 * 0 + 5 * 0 + 8 * 0 + 5.66 * 0 + 8.25 * 0 + 3 * 1 + 6 * 0 + 4.47 * 0 + 6.32 * 0 + 3 * 0 + 4.12 * 0 + 3.61 * 0 + 5.66 * 0 + 2 * 1 + 7.21 * 0 = 19.41$$

- Restrições:

```
Vertice0: x01 + x02 + x03 + x04 + x05 + x06 + x07 + x09 + x0_10 + x0_11 + x0_12 + x0_13 + x0_14 = 1 ;
Vertice1: x01 + x12 + x13 + x14 + x15 + x16 + x17 + x19 + x1_10 + x1_11 + x1_12 + x1_13 + x1_14 = 1 ;
Vertice2: x02 + x12 + x23 + x24 + x25 + x26 + x27 + x29 + x2_10 + x2_11 + x2_12 + x2_13 + x2_14 = 1 ;
Vertice3: x03 + x13 + x23 + x34 + x35 + x36 + x37 + x39 + x3_10 + x3_11 + x3_12 + x3_13 + x3_14 = 1 ;
Vertice4: x04 + x14 + x24 + x34 + x45 + x46 + x47 + x49 + x4_10 + x4_11 + x4_12 + x4_13 + x4_14 = 1 ;
Vertice5: x05 + x15 + x25 + x35 + x45 + x56 + x57 + x59 + x5_10 + x5_11 + x5_12 + x5_13 + x5_14 = 1 ;
Vertice6: x06 + x16 + x26 + x36 + x46 + x56 + x67 + x69 + x6_10 + x6_11 + x6_12 + x6_13 + x6_14 = 1 ;
Vertice7: x07 + x17 + x27 + x37 + x47 + x57 + x67 + x79 + x7_10 + x7_11 + x7_12 + x7_13 + x7_14 = 1 ;
Vertice9: x09 + x19 + x29 + x39 + x49 + x59 + x69 + x79 + x9_10 + x9_11 + x9_12 + x9_13 + x9_14 = 1 ;
Vertice10: x0_10 + x1_10 + x2_10 + x3_10 + x4_10 + x5_10 + x6_10 + x7_10 + x9_10 + x10_11 + x10_12 + x10_13 + x10_14 = 1 ;
Vertice11: x0_11 + x1_11 + x2_11 + x3_11 + x4_11 + x5_11 + x6_11 + x7_11 + x9_11 + x10_11 + x11_12 + x11_13 + x11_14 = 1 ;
Vertice12: x0_12 + x1_12 + x2_12 + x3_12 + x4_12 + x5_12 + x6_12 + x7_12 + x9_12 + x10_12 + x11_12 + x12_13 + x12_14 = 1 ;
Vertice13: x0_13 + x1_13 + x2_13 + x3_13 + x4_13 + x5_13 + x6_13 + x7_13 + x9_13 + x10_13 + x11_13 + x12_13 + x13_14 = 1 ;
Vertice14: x0_14 + x1_14 + x2_14 + x3_14 + x4_14 + x5_14 + x6_14 + x7_14 + x9_14 + x10_14 + x11_14 + x12_14 + x13_14 = 1 ;
```

Com os resultados do LP Solve:

Vértice 0:  $1 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 = 1$

Vértice 1:  $1 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 = 1$

Vértice 2:  $0 + 0 + 0 + 0 + 1 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 = 1$

Vértice 3:  $0 + 0 + 0 + 1 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 = 1$

Vértice 4:  $0 + 0 + 0 + 1 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 = 1$

Vértice 5:  $0 + 0 + 1 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 = 1$

Vértice 6:  $0 + 0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 + 0 + 0 + 0 + 0 = 1$

Vértice 7:  $0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 = 1$

Vértice 9:  $0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 + 0 + 0 + 0 + 0 + 0 = 1$

Vértice 10:  $0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 + 0 + 0 = 1$

Vértice 11:  $0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 + 0 + 0 = 1$

Vértice 12:  $0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 1 = 1$

Vértice 13:  $0 + 0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 + 0 + 0 + 0 + 0 = 1$

Vértice 14:  $0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 = 1$

- O valor da distância total percorrida é o mínimo possível de acordo com todas as restrições. De acordo com o percurso escolhido (figura 11), obtém-se o total mínimo de 101.41 UD (soma de todas as arestas).

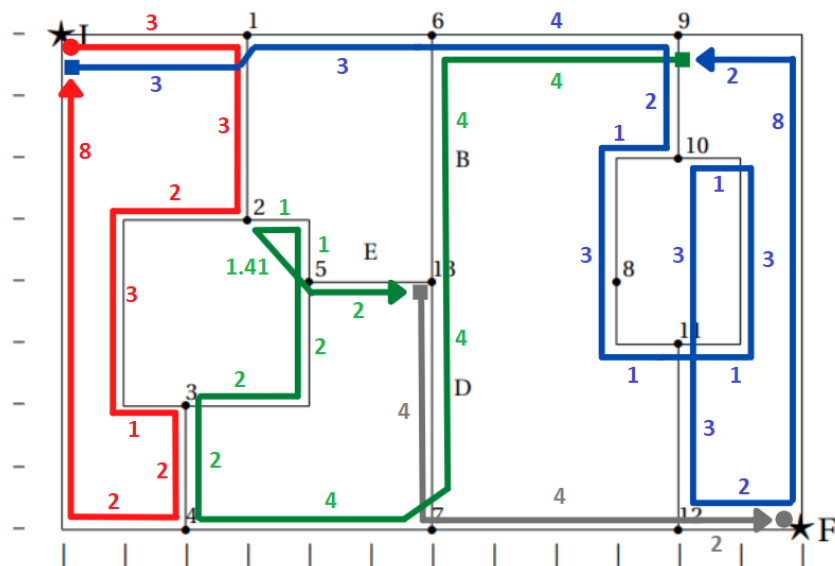


Figura 11 - Caminho ótimo escolhido com as respectivas distâncias

- A distância total percorrida pelo drone é admissível pois, 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).

*distancia total percorrida > soma arestas grafo inicial*  $\rightarrow 101.41 > 82$

## 7. Conclusão

Para concluir, de forma a demonstrar que, qualquer que seja o caminho escolhido que incorpore os novos emparelhamentos, a distância total é constante (101.41 UD), foi selecionado um novo caminho (figura 12) no qual a distância mínima total se mantém a mesma, pois, independentemente do caminho escolhido, a distância total será sempre a soma do resultado obtido no LP Solve (distância mínima “extra” a percorrer) com a distância de todas as arestas, passando pelas mesmas uma única vez.

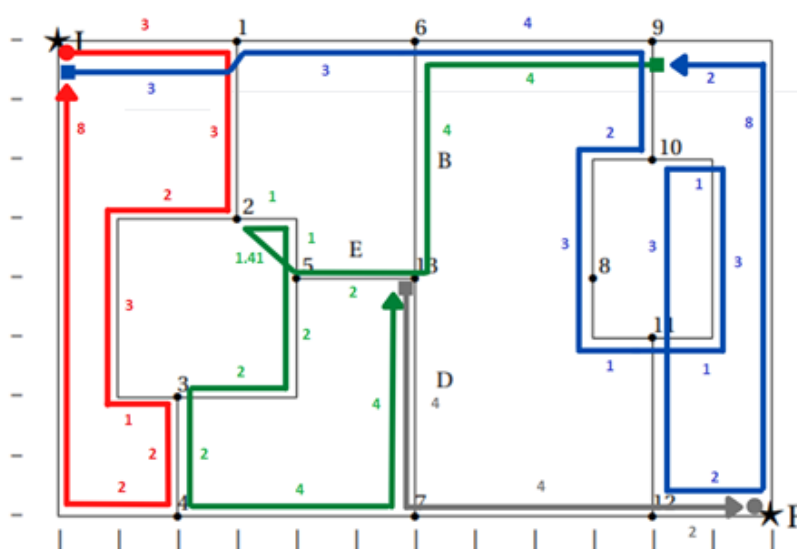


Figura 12 - Caminho ótimo alternativo com as respectivas distâncias

O exemplo alternativo ilustrado acima, tal como na figura 8, é constituído pelo seguinte percurso:

**Vermelho** → **Azul** → **Verde** → **Cinzento**

Concluindo, com este trabalho foi implementada uma solução para o problema do Carteiro Chinês fazendo uso da programação linear, permitindo uma melhor integração com a disciplina.