

**Instituto Politécnico de Beja**  
**Escola Superior de Tecnologia e Gestão**  
**Licenciatura de Engenharia Informática**  
**Época de Finalistas**

**Projeto de Matemática Computacional**

Elaborado por:

Paulo António Luís, nº 17359

Docente:

Maria Teresa de Abreu Luís Godinho

Ano letivo 2020/2021

## Índice

1. Introdução .....	3
2. Comparação e semelhanças problema da Época Normal e de Recurso .....	4
Comparação do problema de Ep. Normal e de Recurso.....	4
Diferença e semelhança entre ambos .....	4
3. Apresentação do problema .....	5
Problema do emparelhamento máximo.....	6
Possível solução .....	7
5. Construção da Heurística .....	9
5.1 Heurística <i>Greedy</i> .....	9
Emparelhamento peso máximo .....	10
Implementação em Matlab .....	11
Eliminar os pares.....	12
Resultados .....	12
Conclusões .....	14

## Índice de figuras

Figura 1 Grafo representativo das relações de compatibilidade nos atletas .....	6
Figura 2 possível combinação de pares de atletas.....	8
Figura 3 Algoritmo Greedy pseudo-código .....	9
Figura 4 código desenvolvido em Matlab .....	11
Figura 5 função delete implementada no Matlab .....	12
Figura 6 leitura da matriz no Matlab .....	12
Figura 7 Solução obtida no Matlab .....	13

## Índice de tabelas

Tabela 1 Registo de pedaladas feitos pelos atletas nas competições em 2020/2021.....	5
Tabela 2 Registo das incompatibilidades entre os atletas .....	5
Tabela 3 Soma das classificações dos ciclistas .....	10
Tabela 4 resultado ciclistas selecionados .....	13

## **1. Introdução**

Neste trabalho pretende-se mostrar compreensão de emparelhamento máximo proposto e à vontade na modelação e resolução do mesmo utilizando programação linear inteira e métodos heurísticos.

O presente relatório encontra-se organizado na seguinte forma: na secção 2 apresenta-se comparação entre o problema de Emparelhamento Máximo e de Peso Máximo, e porquê que o problema em causa pode ser abordado como tal; na secção 3 descreve-se um problema á escolha, na 4 sessão a formulação em Programação Linear Inteira e os possíveis resultados do problema descrito; na secção 5 descreve-se a resolução do problema recorrendo a técnicas heurísticas. Finalmente, na secção 6 são apresentadas as conclusões relativas à elaboração do presente trabalho.

Em anexo a este relatório encontram-se os ficheiros correspondentes à implementação e resolução do problema e no *MATLAB*.

## **2. Comparação e semelhanças problema da Época Normal e de Recurso**

### **Comparação do problema de Ep. Normal e de Recurso**

Ambos os problemas apresentados são diferentes, pois o primeiro trabalho “de época normal” é considerado emparelhamento máximo, apresentou-se uma solução que máximo e procurou-se agrupar par de motoristas “pendura e motorista principal” sem o peso nas arestas. Por sua vez, o de Época. de Recurso é um emparelhamento de peso máximo, porque para selecionar o par “de motorista e pendura” Este problema pode ser considerado um problema de emparelhamento de peso máximo, pois precisa-se de formar pares para realizar viagens de transporte de mercadorias, de acordo com cada máximo das classificações obtida entre a combinação dos motoristas.

### **Diferença e semelhança entre ambos**

O problema de Época normal pode ser considerado um problema de emparelhamento, pois precisa-se de formar pares para realizar viagens de transporte de mercadorias, que inclua duas pessoas “o pendura e o motorista principal”, essa combinação é problema típico de emparelhamento máximo. Já o de Época de Recurso é emparelhamento de peso máximo além de emparelhar dois motoristas, a classificação do par pesa sobre a escolha.

A principal diferença entre ambos está no resultado da classificação da avaliação psicológica “do motorista A com o pendura A” pesa na escolha dos pares.

A semelhança está na capacidade de ambos problemas poderem formar par de nós e combinar dois elementos (x e y, Motorista e pendura), essa é uma característica típica de problemas de emparelhamento, eles também tentam encontrar o maior número possível de arestas a combinar e acredito que ambos os problemas são de gráficos bipartido.

[illegible]

10									0	1
11										0

Pretende-se determinar o número máximo de grupos ou pares que é possível formar sem juntar no mesmo grupo atletas incompatíveis. Este problema é um exemplo típico do problema de emparelhamento máximo.

Para modelar o problema começamos por construir o grafo de compatibilidade. Na Figura 1 representa o grafo das relações de compatibilidades nos atletas, as arestas as ligações compatíveis.

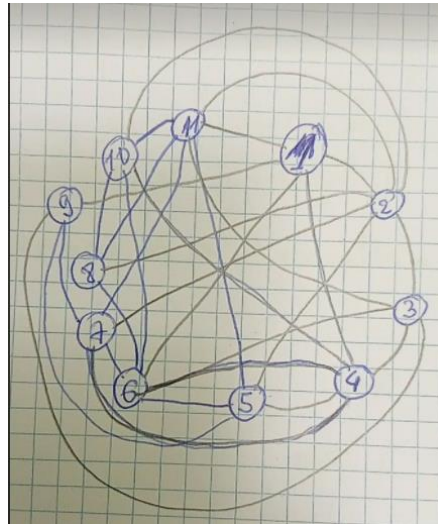


Figura 1 Grafo representativo das relações de compatibilidade nos atletas

### Modelização

Criamos um grafo,  $G = (V, E)$  onde:

$V = \{\text{Ciclistas}\}$ ,  $E = \{(x, y)\}$

$V = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$  e  $E = \{\{1, 2\}, \{1, 4\}, \{1, 6\}, \{1, 9\}, \{1, 11\}, \{2, 3\}, \{2, 5\}, \{2, 7\}, \{2, 8\}, \{2, 10\}, \{2, 11\}, \{3, 4\}, \{3, 6\}, \{3, 9\}, \{3, 11\}, \{4, 5\}, \{4, 6\}, \{4, 7\}, \{4, 10\}, \{5, 6\}, \{5, 9\}, \{5, 11\}, \{6, 7\}, \{6, 8\}, \{6, 10\}, \{7, 9\}, \{7, 11\}, \{8, 9\}, \{8, 10\}, \{8, 11\}, \{9, 10\}, \{9, 11\}, \{10, 11\}\}$

Qual seria uma solução possível? A solução do problema enunciado será o emparelhamento máximo em  $G$ .

### Problema do emparelhamento máximo

Pretendemos determinar o emparelhamento máximo em  $G$ . Para tal, vamos construir e resolver uma formulação em Programação Linear Inteira do problema. Sejam então  $x_{ij}$ , variáveis binárias tais que:

$$x_{\{i,j\}} = \begin{cases} 1 & \text{se a aresta } \{i,j\} \text{ est\u00e1 no emparelhamento} \\ 0 & \text{caso contr\u00e1rio} \end{cases} \quad (\{i,j\} \in E).$$

Uma solu\u00e7\u00e3o deste sistema de inequa\u00e7\u00f5es que permite combinar todos os atletas, para que o objetivo seja cumprido sem que haja pares de atletas incompat\u00edveis e se maximiza o n\u00famero de pares. Usar este conjunto de vari\u00e1veis, podemos formular o problema enunciado como:

$$\begin{aligned} \text{Max } Z = & x_{(1,2)} + x_{(1,4)} + x_{(1,6)} + x_{(1,9)} + x_{(1,11)} + x_{(2,3)} + x_{(2,5)} + x_{(2,7)} + \\ & x_{(2,8)} + x_{(2,10)} + x_{(2,11)} + x_{(3,4)} + x_{(3,6)} + x_{(3,9)} + x_{(3,11)} + x_{(4,5)} + x_{(4,6)} + x_{(4,7)} + x_{(4,10)} + \\ & x_{(5,6)} + x_{(5,9)} + x_{(5,11)} + x_{(6,7)} + x_{(6,8)} + x_{(6,10)} + x_{(7,9)} + x_{(7,11)} + x_{(8,9)} + x_{(8,10)} + \\ & x_{(8,11)} + x_{(9,10)} + x_{(9,11)} + x_{(10,11)} \end{aligned}$$

Sujeito a.:

$$\text{Ciclista 1: } x_{(1,2)} + x_{(1,4)} + x_{(1,6)} + x_{(1,9)} + x_{(1,11)} \leq 1$$

$$\text{Ciclista 2: } x_{(1,2)} + x_{(2,3)} + x_{(2,5)} + x_{(2,7)} + x_{(2,8)} + x_{(2,10)} + x_{(2,11)} \leq 1$$

$$\text{Ciclista 3: } x_{(3,2)} + x_{(3,4)} + x_{(3,6)} + x_{(3,9)} + x_{(3,11)} \leq 1$$

$$\text{Ciclista 4: } x_{(4,3)} + x_{(4,5)} + x_{(4,6)} + x_{(4,7)} + x_{(4,10)} \leq 1$$

$$\text{Ciclista 5: } x_{(4,5)} + x_{(5,6)} + x_{(5,9)} + x_{(5,11)} \leq 1$$

$$\text{Ciclista 6: } x_{(6,1)} + x_{(6,3)} + x_{(6,4)} + x_{(6,5)} + x_{(6,7)} + x_{(6,8)} + x_{(6,10)} \leq 1$$

$$\text{Ciclista 7: } x_{(7,2)} + x_{(7,4)} + x_{(7,6)} + x_{(7,9)} + x_{(7,11)} \geq 1$$

$$\text{Ciclista 8: } x_{(8,2)} + x_{(8,6)} + x_{(8,9)} + x_{(8,10)} + x_{(8,11)} \leq 1$$

$$\text{Ciclista 9: } x_{(9,1)} + x_{(9,3)} + x_{(9,5)} + x_{(9,7)} + x_{(9,8)} + x_{(9,10)} + x_{(9,11)} \leq 1$$

$$\text{Ciclista 10: } x_{(10,2)} + x_{(10,4)} + x_{(10,6)} + x_{(10,8)} + x_{(10,9)} + x_{(10,11)} \leq 1$$

$$\text{Ciclista 11: } x_{(11,1)} + x_{(11,2)} + x_{(11,3)} + x_{(11,5)} + x_{(11,7)} + x_{(11,8)} + x_{(11,9)} + x_{(11,10)} \leq 1$$

$$x_{(1,2)}; x_{(1,4)}; x_{(1,6)}; x_{(1,9)}; x_{(1,11)}; x_{(2,3)}; \dots + x_{(10,11)} \in \{0, 1\}$$

A fun\u00e7\u00e3o objetiva (Max Z) garante que \u00e9 maximizado os n\u00fameros de pares poss\u00edveis.

### **Poss\u00edvel solu\u00e7\u00e3o**

Na formula\u00e7\u00e3o anterior, a fun\u00e7\u00e3o objetivo representa n\u00famero total de arestas a incluir. Por sua vez, as restri\u00e7\u00f5es t\u00e9cnicas asseguram que em cada v\u00e9rtice incide, m\u00e1ximo uma aresta, garantido que a solu\u00e7\u00e3o n\u00e3o inclui arestas adjacentes. As restri\u00e7\u00f5es l\u00f3gicas definem as vari\u00e1veis como vari\u00e1veis bin\u00e1rias.

Resolvendo o modelo apresentado, obtemos  $x_{\{1,10\}} = x_{\{5,7\}} = x_{\{9,11\}} = x_{\{2,4\}} = x_{\{8,3\}} = 1$ . A partir deste resultado, conclu\u00edmos que o emparelhamento obtido \u00e9 um emparelhamento perfeito, pelo que \u00e9 poss\u00edvel formar os grupos respeitando todas as incompatibilidades.

5 Pérez - 7 Botero
9 Blasco - 11 Cuenca
2 Gomes - 4 Serra
8 Bolos - 3 Gomis

Ou seja, o atleta n.º 6 - L. Blanco é o desportista não selecionado.

A Figura 2 representa uma possível solução os resultados:

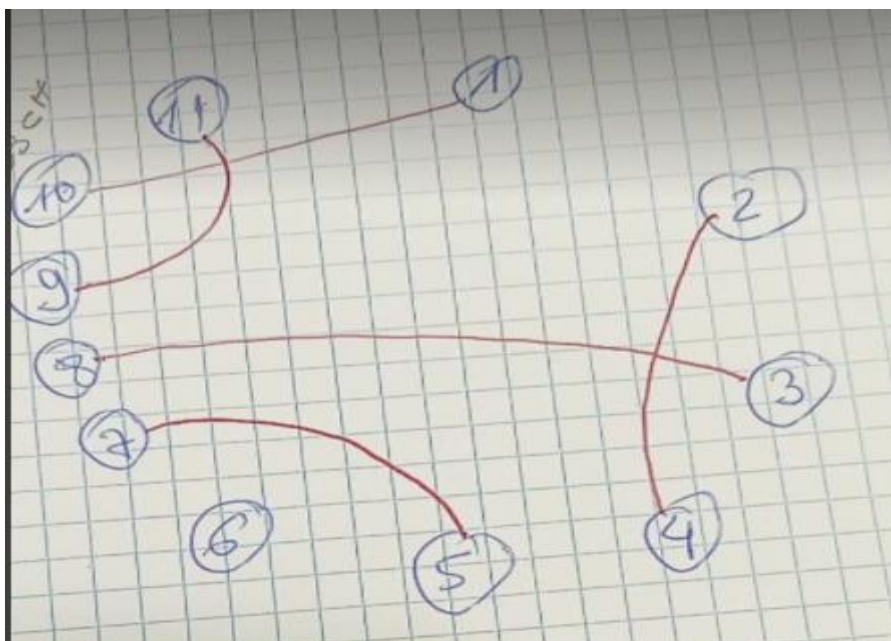


Figura 2 possível combinação de pares de atletas



## 5. Construção da Heurística

Para contrariar o facto de algoritmos como o Simplex, cálculos a mão, levarem muito tempo a calcular a solução ótima, recorrem-se a técnicas heurísticas que encontram uma solução aceitável, em tempo reduzido. Contudo, com a utilização de técnicas heurísticas perdem-se integridade e precisão, visto que a solução encontrada, pode não coincidir com a solução ótima.

Anteriormente, para resolver o problema proposto, foi feita formulação do problema em programação linear inteira. Contudo, quando tratamos problemas da “vida real”, a dimensão dos dados pode tornar a resolução dos problemas, através de alguns algoritmos ou manual, muito demorada. Para evitar isso neste caso usou-se uma heurística construtiva.

### 5.1 Heurística *Greedy*

A heurística construtiva é baseada no algoritmo *Greedy*, em que a cada iteração é escolhida a melhor solução com o objetivo de no final ser encontrada a melhor solução global. É construtiva dado que, no início o conjunto solução encontra-se vazio e a cada iteração é adicionado um elemento a esse conjunto.

A partir deste código representado na Figura 3 começamos a entender o que é necessário fazer para a heurística deste problema e poder começar a trabalhar com o programa *MATLAB* para encontrar uma solução.

O algoritmo de *Greedy* pode ser representado pelo seguinte código:

```
procedure GREEEDY
begin
    while pontos contains at least 1 pair of motoristas
    do begin
        Chose the hightest score pair (p1,pj) E pontos;
        Add (pi,pj) to solucao;
        Delete pi,pj from pontos
    end
end;
```

Figura 3 Algoritmo *Greedy* pseudocódigo

### **Emparelhamento peso máximo**

Para o problema apresentado “em **Apresentação do problema**”, na primeira parte do trabalho, a compatibilidade de cada ciclista era definida como um parâmetro binário que tomava o valor 1 se a diferença ter os valores das pedaladas feita pelos atletas for maior de 40 pedaladas, e o valor 0 caso contrário.

Na segunda fase deste trabalho pretende-se identificar os pares de atletas que se obtêm quando se aplicam diferentes critérios de forma que os pares que se obtêm não sejam repetitivos.

Os novos critérios a serem aplicados são: Apenas deve ser formado 5 pares de ciclistas, deve ser dado privilégio aos atletas que juntos tem melhor classificação. Para isso os especialistas apresentaram na Tabela 1 onde são definidos o resultado da soma das pedaladas dos atletas.

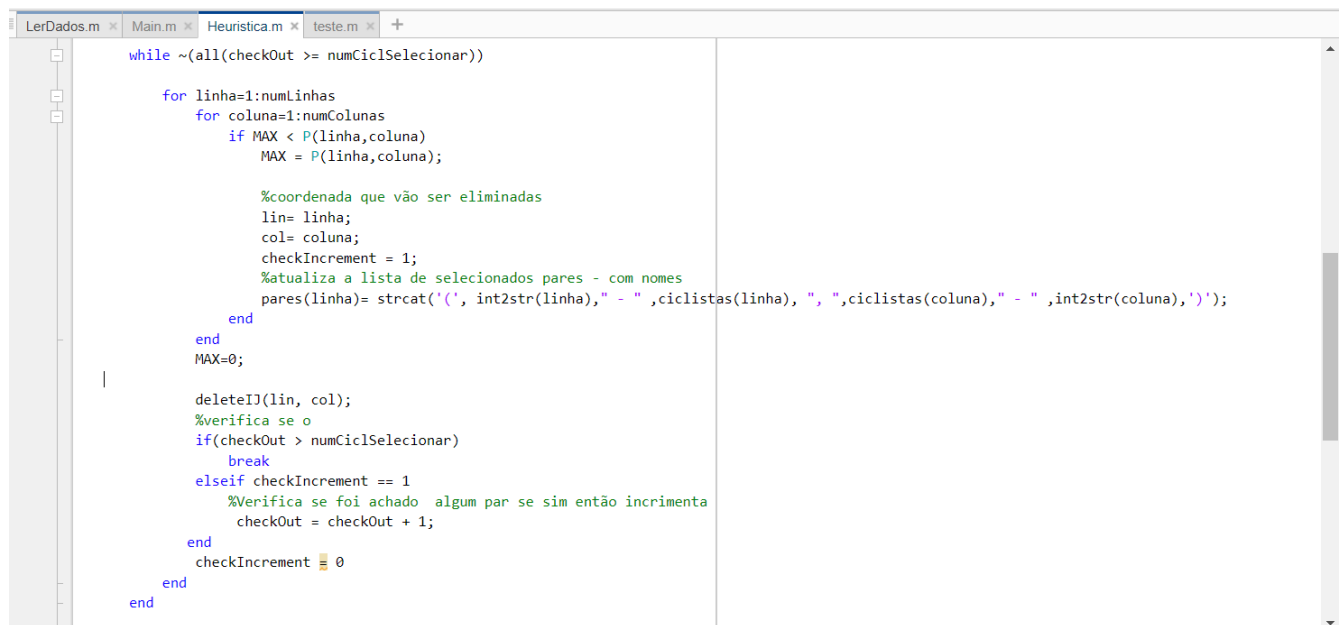
Ciclista (valor das pedaladas)	1 - Ciclista	2 - Ciclista	3 - Ciclista	4 - Ciclista	5 - Ciclista	6 - Ciclista	7 - Ciclista	8 - Ciclista	9 - Ciclista	10 - Ciclista	11 - Ciclista
1 - Ciclista	-1	1170	1100	1150	1105	1210	1100	1110	1190	1080	1175
2 - Ciclista	1170	-1	1170	1220	1175	1280	1170	1180	1260	1150	1245
3 - Ciclista	1100	1170	-1	1150	1105	1210	1100	1110	1190	1080	1175
4 - Ciclista	1150	1220	1150	-1	1155	1260	1150	1160	1240	1130	1225
5 - Ciclista	1105	1175	1105	1155	-1	1215	1105	1115	1195	1085	1180
6 - Ciclista	1210	1280	1210	1260	1215	-1	1210	1220	1300	1190	1285
7 - Ciclista	1100	1170	1100	1150	1105	1210	-1	1110	1190	1080	1175
8 - Ciclista	1110	1180	1110	1160	1115	1220	1110	-1	1200	1090	1185
9 - Ciclista	1190	1260	1190	1240	1195	1300	1190	1200	-1	1170	1265
10 - Ciclista	1080	1150	1080	1130	1085	1190	1080	1090	1170	-1	1155
11 - Ciclista	1175	1245	1175	1225	1180	1285	1175	1185	1155	1155	-1

*Tabela 3 Soma das classificações dos ciclistas*

## Implementação em Matlab

Resumidamente como algoritmo de *Greedy* sugere ir buscar o maior número momentâneo, e após o obter, irá eliminar as linhas e colunas ao qual os ciclistas pertencem, pois formaram um par. Este processo irá repetir até não ser mais possível formar pares. No final irá se obter os pares que este algoritmo formou.

Para tal foi necessário de criar um pseudocódigo que irá servir como base para a codificação em Matlab. Na Figura 4 apresenta-se o desenvolvimento do código que procura na linha e na coluna o maior valor guarda-o na lista de pares seleccionados, de seguida atualiza a variável *MAX* e o *checkout*, faz o *break* do *for* quando já tiver pares de atletas, no *while* verifica se já estão seleccionados todos os atletas se sim então sai do *while*.



```
LerDados.m x Main.m x Heuristica.m x teste.m x +
while ~(all(checkOut >= numCiclSelecionar))
    for linha=1:numLinhas
        for coluna=1:numColunas
            if MAX < P(linha,coluna)
                MAX = P(linha,coluna);

                %coordenada que vão ser eliminadas
                lin= linha;
                col= coluna;
                checkIncrement = 1;
                %atualiza a lista de seleccionados pares - com nomes
                pares(linha)= strcat('(', int2str(linha), " - ",ciclistas(linha), " ",ciclistas(coluna)," - ",int2str(coluna),')');
            end
        end
        MAX=0;
    |
    deleteIJ(lin, col);
    %verifica se o
    if(checkOut > numCiclSelecionar)
        break
    elseif checkIncrement == 1
        %Verifica se foi achado algum par se sim então incrementa
        checkOut = checkOut + 1;
    end
    checkIncrement = 0
end
end
```

Figura 4 código desenvolvido em Matlab

## Eliminar os pares

A função *deleteIJ* – tem o papel de eliminar as linhas e as colunas já seleccionadas. Com o *try catch* é verificado se é apresentado algum erro, no *for* de até 11 são atualizadas ou removidas as linhas e as colunas dos parâmetros *i* e *j* na matriz principal. A Figura 5 pode se observa o método desenvolvido em *Matlab*

```
%Função para eliminar linha e coluna já seleccionadas
function deleteIJ(i, j)
    try
        global P;
        for l=1:11
            P(i, l) = -1;
            P(l, i) = -1;

            P(j, l) = -1;
            P(l, j) = -1;
        end
    catch
        fprintf('Erro ao apagar Linha e Coluna!\n');
    end
end
```

Figura 5 função delete implementada no Matlab

## Resultados

A Figura 6 representa a matriz principal que é pretendido minimizar o numero de pares.

```
Menu
A - Seleccionar ficheiro de dados
B - Correr Heurística
T - Terminar o programa
Escolha uma opcao:
A
Existem 1 ficheiros.
newmatriz.mat
Nome do ficheiro a ler:
new
Ficheiro lido com sucesso.
Número de linhas lidas: 11
Número de colunas lidas: 11
-1      1170      1100      1150      1105      -1      1100      1110      1190      1080      1175
1170      -1      1170      1220      1175      -1      1170      1180      1260      1150      1245
1100      1170      -1      1150      1105      -1      1100      1110      1190      1080      1175
1150      1220      1150      -1      1155      -1      1150      1160      1240      1130      1225
1105      1175      1105      1155      -1      -1      1105      1115      1195      1085      1180
-1      -1      -1      -1      -1      -1      1210      1220      1300      1190      1285
1100      1170      1100      1150      1105      1210      -1      1110      1190      1080      1175
1110      1180      1110      1160      1115      1220      1110      -1      1200      1090      1185
1190      1260      1190      1240      1195      1300      1190      1200      -1      1170      1265
1080      1150      1080      1130      1085      1190      1080      1090      1170      -1      1155
1175      1245      1175      1225      1180      1285      1175      1185      1155      1155      -1
```

Figura 6 leitura da matriz no Matlab

É importante referir que antes de executar o código, é necessário ler os ficheiros que contêm a matriz de cobertura e a matriz auxiliar.

Depois de executar o código podemos ver que a solução apresenta cinco combinações com exigido no problema, essa combinação teve o peso consoante as classificações ou número de pedaladas dos ciclistas. Assim sendo, foram apenas seleccionados 5 atletas para os jogos olímpicos, e o atleta "M. Basto" fica de fora.

```

Pares Seleccionados
(1 - A. Ramos, L. Blanco - 6)
(2 - J. Gomes, J. Blasco - 9)
(3 - S. Gomis, J. Cuenca - 11)
(4 - J. Serna, R. Bolos - 8)
(5 - R. Pires, S. Botero - 7)
\n\n
Menu
A - Seleccionar ficheiro de dados
B - Correr Heurística
T - Terminar o programa
Escolha uma opcao:

```

Figura 7 Solução obtida no Matlab

Os pares escolhidos pelo algoritmo GREEDY são:  $solucao = X_{(1,6)}, X_{(2,9)}, X_{(3,11)}, X_{(4,8)}, X_{(5,7)}$

Ciclista (valor das pedaladas)	1 - Ciclista	2 - Ciclista	3 - Ciclista	4 - Ciclista	5 - Ciclista	6 - Ciclista	7 - Ciclista	8 - Ciclista	9 - Ciclista	10 - Ciclista	11 - Ciclista
1 - Ciclista						1210					
2 - Ciclista									1260		
3 - Ciclista											1175
4 - Ciclista								1160			
5 - Ciclista							1105				
6 - Ciclista											
7 - Ciclista											
8 - Ciclista											
9 - Ciclista											
10 - Ciclista											
11 - Ciclista											

Tabela 4 resultado ciclistas seleccionados

## Conclusões

Neste trabalho, foi apresentado um problema de emparelhamento máximo e de peso máximo. A principal dificuldade deste tipo de problemas, prende-se com o conseguimento da matriz e interpretação da heurística de *Greedy*, através dos dados. Assim que se obtém a matriz, com a ajuda do a solução encontra-se com relativa facilidade.

A utilização de técnicas heurísticas tem vantagens e desvantagens. A principal vantagem é o facto de exigir muito menos tempo no cálculo da solução, em comparação com o uso de algoritmos como o *Simplex*. Outra vantagem é que pelo facto de as heurísticas não serem muito precisas nem exatas, permitem resolver problemas com um grau elevado de complexidade que de outra forma não seriam possíveis de resolver. Por outro lado, pela mesma razão, as soluções encontradas não são necessariamente as soluções ótimas.