ALGORITMIA AVANÇADA

SPRINT C

TURMA 3NA

GRUPO 74

| 1140858 Carlos Filipe Borges Moutinho
| 1171602 Rui Manuel Castro Marinho
| 1181882 Rafael Martins Soares
| 1181892 Sara Santos Teixeira Silva
| 1181895 Fábio Alves da Silva

ÍNDICE

INTRODUÇÃO

O presente trabalho foi desenvolvido no âmbito da disciplina de Algoritmia Avançada (ALGAV), onde foi pedido o desenvolvimento do planeamento de contatos numa rede social. Para a mesma, foi utilizada a linguagem de programação PROLOG e utilizados métodos de pesquisa lecionados durante o semestre.

Neste relatório será efetuada o estudo das seguintes UC's:

- Criação de uma rede à parte com os utilizadores que podem ser alcançados até N ligações a partir de um dado utilizador
- Gerador aleatório de forças
- Adaptação do A* ao problema da determinação do caminho mais forte (máximo de N ligações)
- Implementar a estimativa
- Adaptação do Best First ao problema da determinação do caminho mais forte (máximo de N ligações)
- Adaptação do Primeiro em Profundidade para gerar a melhor solução (já implementado no Sprint anterior) para o máximo de N ligações
- Comparação dos 3 métodos com vários exemplos, comparando tempos de geração da solução e valor da solução gerada
- Implementação da função multicritério que contemple forças de ligação e diferença entre likes e dislikes
- Adaptação dos 3 métodos (Primeiro em Profundidade, Best First e A*) para considerar a função multicritério do ponto anterior
- Comparação dos 3 métodos com vários exemplos e usando a função multicritério

DESENVOLVIMENTO

CRIAÇÃO DE UMA REDE À PARTE COM OS UTILIZADORES QUE PODEM SER ALCANÇADOS ATÉ N LIGAÇÕES A PARTIR DE UM DADO UTILIZADOR

De forma a responder ao requerimentos do sprint C criamos uma nova rede. Abaixo encontra-se 3 imagens recriando um grafo com indicação das forças de ligação, forças de relação e conjunto de tags respetivamente.

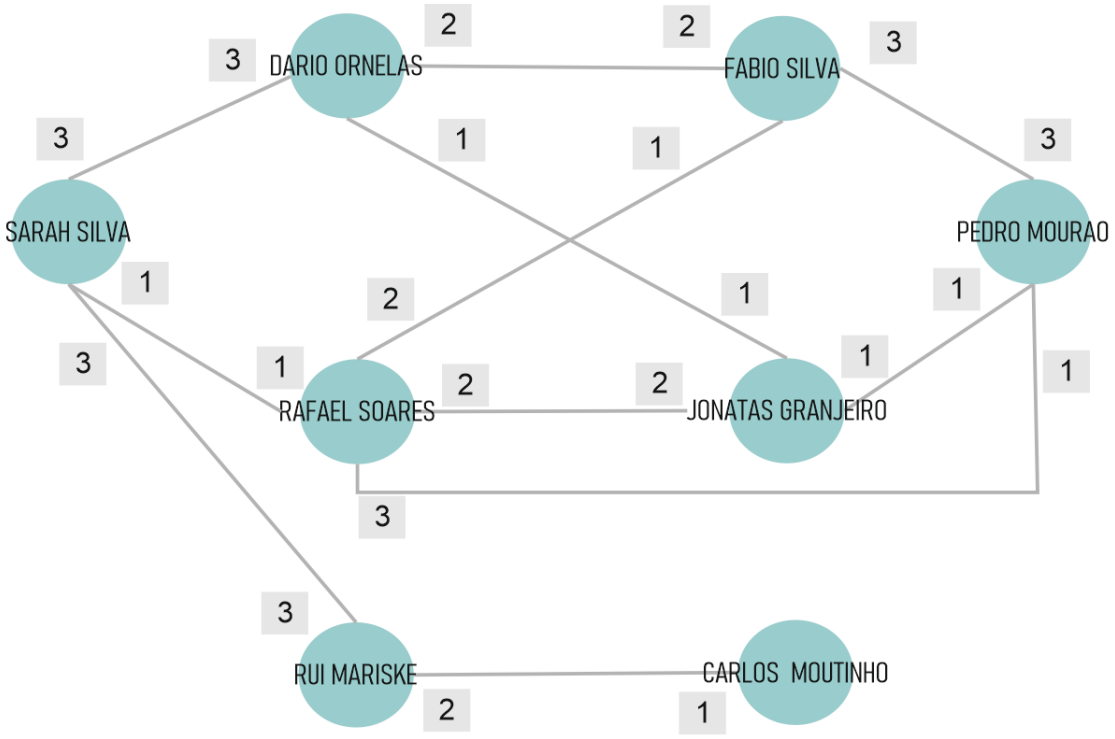


Figura 1 – Grafo com forças de ligação

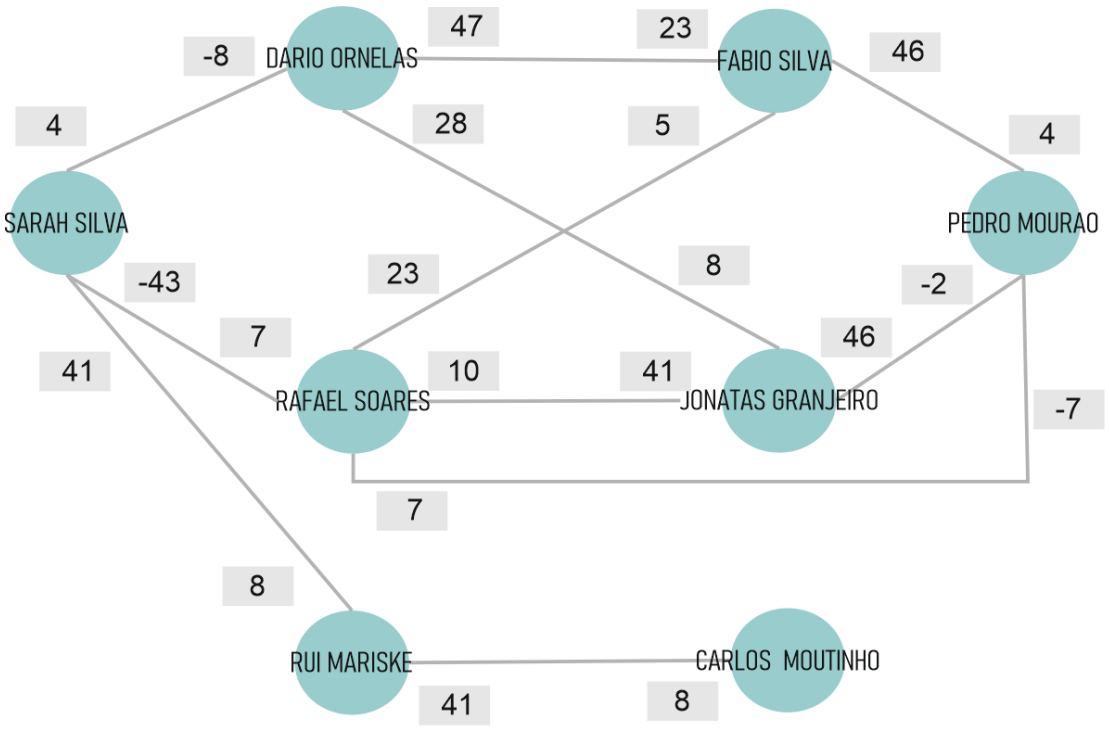


Figura 2 – Grafo com forças de relação.

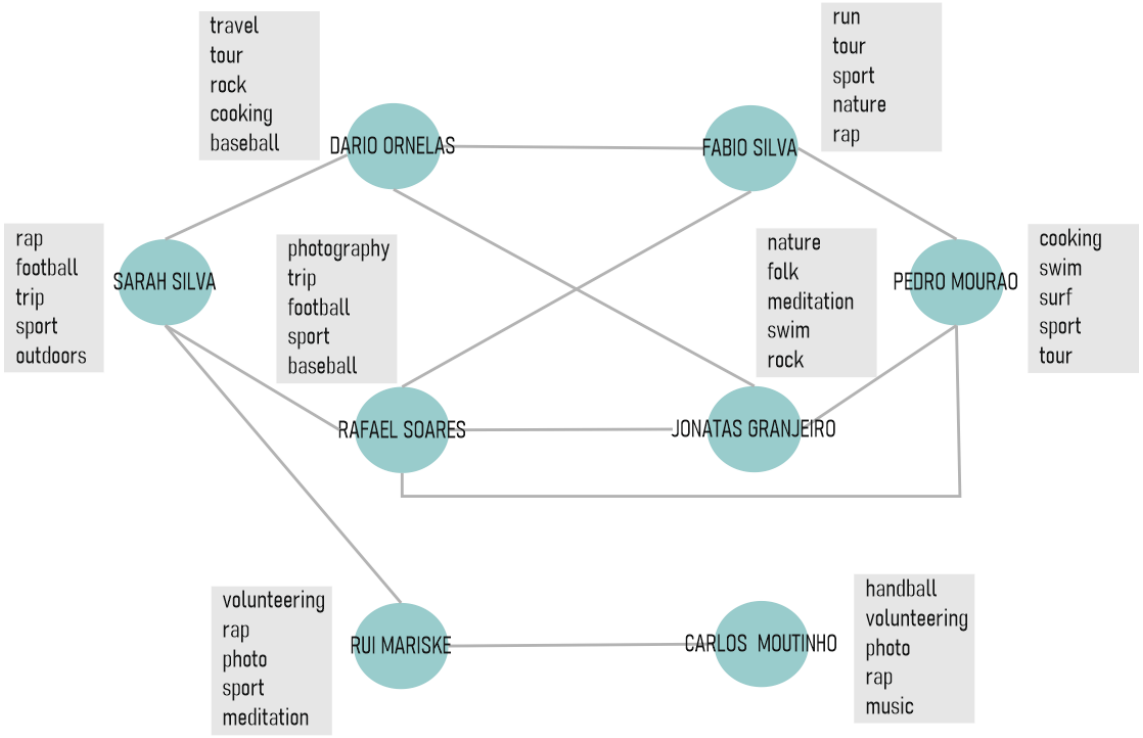


Figura 3 – Grafo com indicação das tags de cada utilizador.

GERADOR ALEATÓRIO DAS FORÇAS

```
% gerenciador de forças

:-dynamic ligacao/4.

generate_ForcasLigacao(Novos):-

        findall((A,B,L,C),
                ( random(1,3,L),
                  ligacao(A,B,_,C),
                  retract(ligacao(_,_,_,_)),
                  asserta(ligacao(A,B,L,C)))
               ,Novos).
```

Exemplo:

Ligação em estudo: *Carlos Moutinho – Rui Mariske*

Base de conhecimento:

```
ligacao('Carlos Moutinho', 'Rui Mariske', 1, 8).
```

```
?- ligacao('Carlos Moutinho', 'Rui Mariske',L,_).
L = 1 .

?- generate_ForcasLigacao(N).
N = [('Carlos Moutinho', 'Rui Mariske', 2, 8), ('Carlos Moutinho', 'Rui Mariske', 2, 8),
 ('Carlos Moutinho', 'Rui Mariske', 2, 8), ('Carlos Moutinho', 'Rui Mariske', 2, 8), ('C
arlos Moutinho', 'Rui Mariske', 2, 8), ('Carlos Moutinho', 'Rui Mariske', 2, 8), ('Carlo
s Moutinho', 'Rui Mariske', ..., ...), ('Carlos Moutinho', ..., ...), (..., ...)|...].

?- ligacao('Carlos Moutinho', 'Rui Mariske',L,_).
L = 2 .
```

Como visto na imagem acima, quando pretendemos verificar qual a força da ligação entre ambos os utilizadores obtemos uma ligação=1.

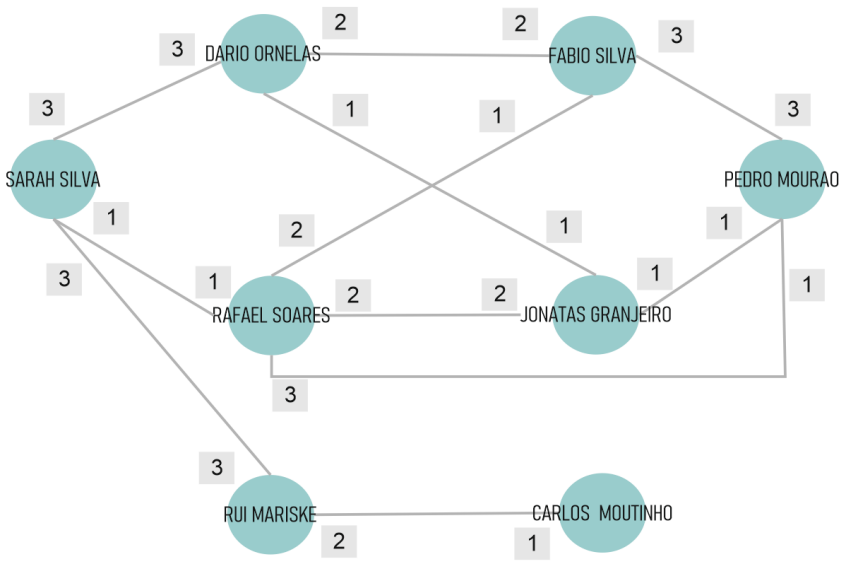
Ao gerarmos novas forças de ligação através do método generate_ForcasLigacao, estamos a mudar o valor das forças de ligação em toda a rede. Para comprovar a eficiência do algoritmo, voltamos a chamar o predicado 'ligação' para verificar que efetivamente o valor da ligação exemplo mudou.

Adaptação do A* ao problema da determinação do caminho mais forte (máximo de N ligações)

Neste código apenas são considerados as forças de ligação e um número de máximo de ligações. No que diz respeito à estimativa, foi colocada a 0.

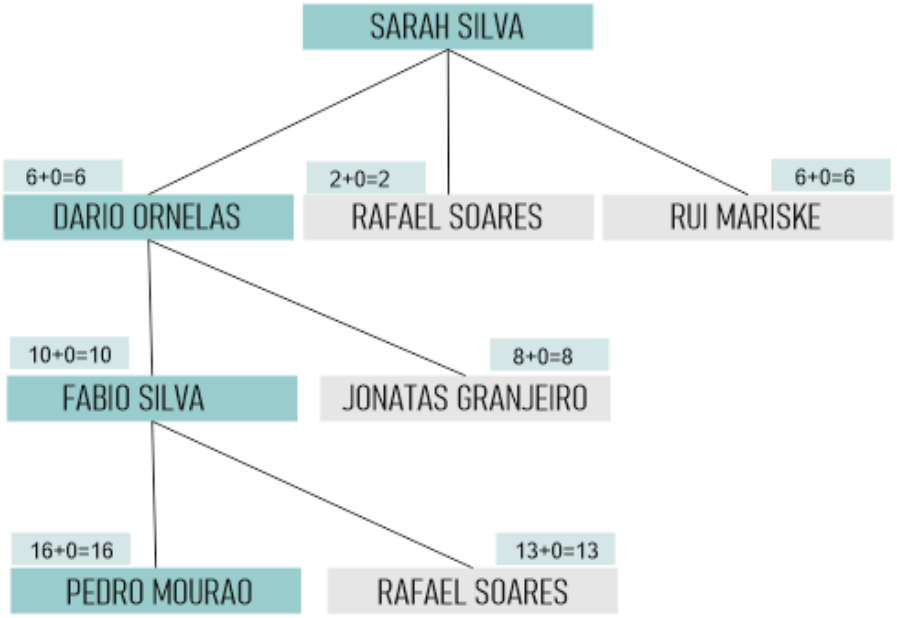
```
aStar_NoEstimation(Orig,Dest,Nivel,Cam,Custo):-
        aStar1_NoEstimation(Dest,[(_,0,[Orig])],Nivel,0,Cam,Custo).

aStar1_NoEstimation(Dest,[(_,Custo,[Dest|T])|_],_,_,Cam,Custo):-
        reverse([Dest|T],Cam).

aStar1_NoEstimation(Dest,[(_,Ca,LA)|Outros],NivelMax,NivelAtual,Cam,Custo):-
        NivelAtual =< NivelMax,
        member(Act,LA),
        length(LA,NumUsers),
        NivelAtual1 is NumUsers-1,
        NivelAtual1 < NivelMax,

        findall((CEX,CaX,[X|LA]),
                (Dest\==Act,
                 (ligacao(Act,X,ForcaX,_),ligacao(X,Act,ForcaY,_)),
                 \+ member(X,LA),
                 CaX is ForcaX + ForcaY + Ca,
                 estimativa0(CaX,NivelMax,NivelAtual1,EstX),
                 CEX is CaX +EstX),
                Novos),

        append(Outros,Novos,Todos),
        %write('Novos='),write(Novos),nl,
        sort(0,  @>, Todos,  TodosOrd),
        write('TodosOrd='),write(TodosOrd),nl,
        aStar1_NoEstimation(Dest,TodosOrd,NivelMax,NivelAtual1,Cam,Custo).

aStar1_NoEstimation(Dest,[(_,_,_)|Outros],NivelMax,NivelAtual,Cam,Custo):-
        aStar1_NoEstimation(Dest,Outros,NivelMax,NivelAtual,Cam,Custo).

estimativa0(_,_,_,0).
```

Para fins de exemplo utilizamos o seguinte grafo:



Teste com máximo de 3 ligações (2 nós intermédios):

```
?- aStar_NoEstimation('Sarah Silva', 'Pedro Mourao',3,Cam,Custo).
Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Pedro Mourao'],
Custo = 16 .
```

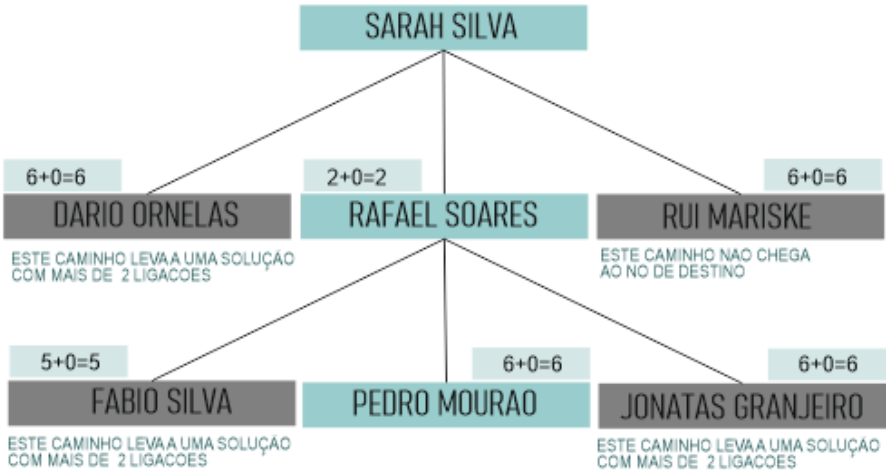


O resultado e a traçagem acima apresentada validam ambos o resultado obtido e o cálculo do custo.

Esta solução apresenta 2 nós intermédios e, de forma a validarmos que o nosso predicado funcione no que diz respeito ao controlo do número de ligações iremos testar a mesma condição indicando como número máximo de ligações o valor 2 e 1.

Teste com máximo de 2 ligações (1 nó intermédio):

```
?- aStar_NoEstimation('Sarah Silva', 'Pedro Mourao',2,Cam,Custo).
Cam = ['Sarah Silva', 'Rafael Soares', 'Pedro Mourao'],
Custo = 6 .
```



Teste com máximo de 1 ligações (0 nós intermédios):

```
?- aStar_NoEstimation('Sarah Silva', 'Pedro Mourao',1,Cam,Custo).
false.
```

Tal como expectável, nenhuma solução é disponibilizada uma vez que não existe nenhuma ligação direta entre o utilizador Orig e Dest.

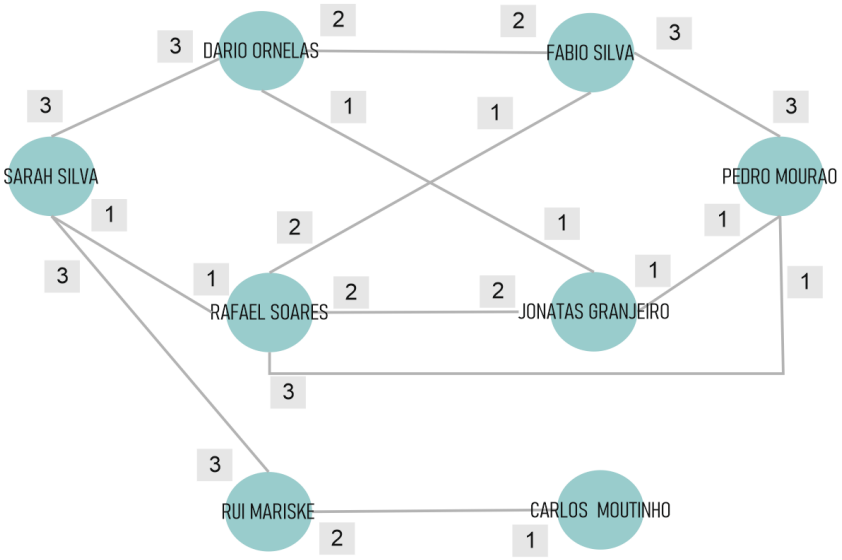
IMPLEMENTAR A ESTIMATIVA NO ALGORITMO A*

Neste exercício implementamos uma estimativa no algoritmo A*.

```
aStar(Orig,Dest,NivelMax,Cam,Custo):-
        get_ForcasLigacao(ListaOrdenadaFL),
        aStar2(Dest,[(_,0,[Orig])],NivelMax,0,ListaOrdenadaFL,Cam,Custo).

aStar2(Dest,[(_,Custo,[Dest|T])|_],_,_,_,Cam,Custo):-
        reverse([Dest|T],Cam).

aStar2(Dest,[(_,Ca,LA)|Outros],NivelMax,NivelAtual,ListaOrdenadaFL,Cam,Custo):-
        NivelAtual =< NivelMax,
        member(Act,LA),
        length(LA,NumUsers),
        NivelAtual1 is NumUsers-1,
        NivelAtual1 < NivelMax,
        nth0(NivelAtual1,ListaOrdenadaFL, ForcaMax),

        findall((CEX,CaX,[X|LA]),
                (Dest\==Act,
                 (ligacao(Act,X,ForcaX,_),ligacao(X,Act,ForcaY,_)),
                 \+ member(X,LA),
                 CaX is ForcaX + ForcaY + Ca,
                 estimativa(ForcaMax,NivelMax,NivelAtual1,EstX),
                 CEX is CaX +EstX),
                Novos),

        append(Outros,Novos,Todos),
        %write('Novos='),write(Novos),nl,
        sort(0, @>, Todos,  TodosOrd),
        %write('TodosOrd='),write(TodosOrd),nl,
        aStar2(Dest,TodosOrd,NivelMax,NivelAtual1,ListaOrdenadaFL,Cam,Custo).

aStar2(Dest,[(_,_,_)|Outros],NivelMax,NivelAtual,ListaOrdenadaFL,Cam,Custo):-
        aStar2(Dest,Outros,NivelMax,NivelAtual,ListaOrdenadaFL,Cam,Custo).


estimativa(ForcaMax,Nivel,NivelAtual,Estimativa):-
        Estimativa is ForcaMax *(Nivel-NivelAtual).


get_ForcasLigacao(ListaOrdenadaFL):-
        findall(ForcaLigacao,
                (ligacao(A,B,FL1,_),ligacao(B,A,FL2,_), ForcaLigacao is FL1+FL2),
                ListaForcasLigacao),
        sort(0,@>=,ListaForcasLigacao, ListaOrdenadaFL).
```

Para o cálculo da estimativa , obtivemos uma lista ordenada de todas as forças de ligação presentes no grafo (método: get_ForcasLigacao). A estimativa é calculada multiplicando a força máxima do grafo com o número de níveis ainda a percorrer (diferença entre o nível máximo imposto e o nível atual). A força máxima do grafo usada no cálculo varia de nível para nível. A lista de todas as forças ordenadas, permite obter as forças de ligação usadas por cada nível , isto é, numa primeira ligação a estimativa é calculada usando o primeiro valor da lista de forças; numa segundo iteração, será usada o segundo valor da lista e por aí a diante.

Para fins de exemplo utilizamos o seguinte grafo:



Teste com máximo de 3 ligações (2 nós intermédios):

```
?- aStar('Sarah Silva', 'Pedro Mourao',3,Cam,Custo).

TodosOrd=[(24,6,[Rui Mariske,Sarah Silva]),(24,6,[Dario Ornelas,Sarah Silva]),(20,2,[Rafael
Soares,Sarah Silva])]


TodosOrd=[(24,6,[Dario Ornelas,Sarah Silva]),(21,9,[Carlos Moutinho,Rui Mariske,Sarah Silva]
),(20,2,[Rafael Soares,Sarah Silva])]


TodosOrd=[(22,10,[Fabio Silva,Dario Ornelas,Sarah Silva]),(21,9,[Carlos Moutinho,Rui Mariske
,Sarah Silva]),(20,8,[Jonatas Granjeiro,Dario Ornelas,Sarah Silva]),(20,2,[Rafael Soares,Sar
ah Silva])]


TodosOrd=[(22,16,[Pedro Mourao,Fabio Silva,Dario Ornelas,Sarah Silva]),(21,9,[Carlos Moutinh
o,Rui Mariske,Sarah Silva]),(20,8,[Jonatas Granjeiro,Dario Ornelas,Sarah Silva]),(20,2,[Rafa
el Soares,Sarah Silva]),(19,13,[Rafael Soares,Fabio Silva,Dario Ornelas,Sarah Silva])]

Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Pedro Mourao'],
Custo = 16 .
```

Teste com máximo de 2 ligações (1 nó intermédios):

```
?- aStar('Sarah Silva', 'Pedro Mourao',2,Cam,Custo).

TodosOrd=[(18,6,[Rui Mariske,Sarah Silva]),(18,6,[Dario Ornelas,Sarah Silva]),(14,2,[Rafael
Soares,Sarah Silva])]


TodosOrd=[(18,6,[Dario Ornelas,Sarah Silva]),(15,9,[Carlos Moutinho,Rui Mariske,Sarah Silva]
),(14,2,[Rafael Soares,Sarah Silva])]


TodosOrd=[(16,10,[Fabio Silva,Dario Ornelas,Sarah Silva]),(15,9,[Carlos Moutinho,Rui Mariske
,Sarah Silva]),(14,8,[Jonatas Granjeiro,Dario Ornelas,Sarah Silva]),(14,2,[Rafael Soares,Sar
ah Silva])]


TodosOrd=[(12,6,[Pedro Mourao,Rafael Soares,Sarah Silva]),(12,6,[Jonatas Granjeiro,Rafael So
ares,Sarah Silva]),(11,5,[Fabio Silva,Rafael Soares,Sarah Silva])]

Cam = ['Sarah Silva', 'Rafael Soares', 'Pedro Mourao'],
Custo = 6 .
```

Neste exercício, sendo exigido um máximo de 2 ligações o caminho Sarah-Dario-Fábio-Pedro não é possível uma vez que possui 2 nós intermédios. O segundo caminho com maior estimativa é Sarah-Rui-Carlos e neste caso também não é possível pelo facto do Carlos não ter mais ligações. Por fim, temos o caminho Sarah-Rafael-Pedro que preenche todos os requisitos e é dado como resposta ao problema.

Teste com máximo de 1 ligações (0 nó intermédios):

```
?- aStar('Sarah Silva', 'Pedro Mourao',1,Cam,Custo).

TodosOrd=[(12,6,[Rui Mariske,Sarah Silva]),(12,6,[Dario Ornelas,Sarah Silva]),(8,2,[Rafael S
oares,Sarah Silva])]

false.
```

Neste teste é pedido um caminho com 1 ligação apenas, o que implica uma ligação direta entre o userOrig e o UserDest. Não havendo esse tipo de ligação no nosso grafo, o resultado é false.

Adaptação do Best First ao problema da determinação do caminho mais forte (máximo de N ligações)

Neste exercício implementamos um máximo de ligações no algoritmo ao BestFirst.

```
bestfs1(Orig,Dest,Nivel,Cam,Custo):-
        bestfs12(Dest,[[Orig]],Nivel,0,Cam,Custo),
        write('Caminho='),write(Cam),nl.

bestfs12(Dest,[[Dest|T]|_],_,_,Cam,Custo):-
        reverse([Dest|T],Cam),
        calcula_custo(Cam,Custo).

bestfs12(Dest,[[Dest|_]|LLA2],NivelMax,NivelAtual,Cam,Custo):-
        !,bestfs12(Dest,LLA2,NivelMax,NivelAtual,Cam,Custo).


bestfs12(Dest,LLA,NivelMax,NivelAtual,Cam,Custo):-
        NivelAtual =< NivelMax,
        LA=[Act|_],
        member1(LA,LLA,LLA1),
        length(LA,NumUsers),
        NivelAtual1 is NumUsers-1,
        ((Act==Dest,!,bestfs12(Dest,[LA|LLA1],NivelMax,NivelAtual,Cam,Custo));
         (NivelAtual1 < NivelMax,
          findall((CX,[X|LA]),
                  (ligacao(Act,X,FLX,_),ligacao(X,Act,FLY,_),
                   CX is FLX+FLY,
                   \+member(X,LA)),Novos),
          Novos\==[],!,
          sort(0,@>=,Novos,NovosOrd),
          retira_custos(NovosOrd,NovosOrd1),
          append(NovosOrd1,LLA1,LLA2),
          write('LLA2='),write(LLA2),nl,
          bestfs12(Dest,LLA2,NivelMax,NivelAtual1,Cam,Custo))).

member1(LA,[LA|LAA],LAA).
member1(LA,[_|LAA],LAA1):-
        member1(LA,LAA,LAA1).

retira_custos([],[]).
retira_custos([(_,LA)|L],[LA|L1]):-
        retira_custos(L,L1).

calcula_custo([Act,X],C):-!,
        ligacao(Act,X,FLx,_),
        ligacao(X,Act,FLy,_),
        C is FLx + FLy.
calcula_custo([Act,X|L],S):-
        calcula_custo([X|L],S1),
        ligacao(Act,X,C,_),
        ligacao(X,Act,C1,_),
        S is S1+C+C1.
```

De forma a controlar o número de ligações controlamos o número de nós na lista do caminho. Consideramos o número de ligações atuais como sendo o número de nós presentes na lista menos 1 (referente ao userOrig) – exemplo: na lista [S-T-E-A] consideramos o nível=3 (3 ligações). Esse valor é então comparado ao nível máximo inserido pelo utilizador.

Teste com máximo de 3 ligações (2 nós intermédios):

```
?- bestfs1('Sarah Silva','Pedro Mourao',3,Cam,Custo).
Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Pedro Mourao'],
Custo = 16 .
```

Teste com máximo de 2 ligações (1 nós intermédios):

```
?- bestfs1('Sarah Silva','Pedro Mourao',2,Cam,Custo).
Cam = ['Sarah Silva', 'Rafael Soares', 'Pedro Mourao'],
Custo = 6 .
```

Teste com máximo de 1 ligações (sem nós intermédios):

```
?- bestfs1('Sarah Silva','Pedro Mourao',1,Cam,Custo).
false.
```

Adaptação do Primeiro em Profundidade para gerar a melhor solução (já implementado no Sprint anterior) para o máximo de N ligações

Neste exercício implementamos um máximo de ligações no algoritmo ao DFS.

```
:-dynamic melhor_sol_maisForte/2.
:-dynamic conta_sol_maisForte/1.


caminho_maisForte2(Orig,Dest,Nivel,LCaminho_maisForte,Forca):-
                    get_time(Ti),
                   (melhor_caminho_maisForte2(Orig,Dest,Nivel);true),
                    retract(melhor_sol_maisForte(LCaminho_maisForte,Forca)),
                    retract(conta_sol_maisForte(NS1)),
                    get_time(Tf),
                    T is Tf-Ti,
                    nl, write('Número de soluções encontradas: '),write(NS1),nl,
                    nl, write('Tempo de geracao da solucao:'),write(T),nl.


melhor_caminho_maisForte2(Orig,Dest,Nivel):-
                    asserta(melhor_sol_maisForte(_,0)),
                    asserta(conta_sol_maisForte(0)),
                    dfs(Orig,Dest,LCaminho),
                    atualiza_melhor_maisForte2(LCaminho,Nivel),
                    fail.

atualiza_melhor_maisForte2(LCaminho,Nivel):-
                    retract(conta_sol_maisForte(NS)),
                    NS1 is NS+1,
                    asserta(conta_sol_maisForte(NS1)),
                    melhor_sol_maisForte(_,N),
                    calculateStrength2(LCaminho,[],SumStrength) ,
                    length(LCaminho,C),
                    %nl, write(LCaminho), nl,
                    C1 is C-1,
                    C1=<Nivel, !,
                    write(SumStrength),
                    nl, write("Caminho: "), write(LCaminho), nl,
                    SumStrength>N,
                    retract(melhor_sol_maisForte(_,_)),
                    asserta(melhor_sol_maisForte(LCaminho,SumStrength)).


calculateStrength2([_],Temp,SumStrength):-
        sumList(Temp,SumStrength),!.

calculateStrength2([A,B|T], Temp, Strength):-
                    ligacao(A,B,C1,_),
                    ligacao(B,A,C2,_),
                    Soma is C1+C2,

                    removeElement([A,B|T],A,List),

                    calculateStrength2(List,[Soma|Temp],Strength).
```

De forma a validar um número máximo de ligações, a cada caminho verificamos o tamanho da lista e subtraímos 1 (relativo ao userOrig). Efetuando essa subtração estamos a auferir o número de ligações do caminho em estudo. Este valor é depois comparado com o valor máximo de ligações imposto pelo utilizador.

COMPARAÇÃO DOS 3 MÉTODOS COM VÁRIOS EXEMPLOS, COMPARANDO TEMPOS DE GERAÇÃO DA SOLUÇÃO E VALOR DA SOLUÇÃO GERADA

NÍVEL 2

	TEMPO	FORÇA DE LIGAÇÃO
A*	0.00	6
BestFirst	0.00	6
Primeiro em Profundidade	0.000999	6

```
?- aStar('Sarah Silva','Pedro Mourao',2,Cam,Custo).

Tempo de geracao da solucao:0.0
Cam = ['Sarah Silva', 'Rafael Soares', 'Pedro Mourao'],
Custo = 6 .

?- bestfs1('Sarah Silva', 'Pedro Mourao',2,Cam,Custo).

Tempo de geracao da solucao:0.0
Cam = ['Sarah Silva', 'Rafael Soares', 'Pedro Mourao'],
Custo = 6 .

?- caminho_maisForte2('Sarah Silva', 'Pedro Mourao',2,Cam,Custo).

Tempo de geracao da solucao:0.0009989738464355469
Cam = ['Sarah Silva', 'Rafael Soares', 'Pedro Mourao'],
Custo = 6.
```

NÍVEL 3

	TEMPO	FORÇA DE LIGAÇÃO
A*	0.001004	16
BestFirst	0.00	16
Primeiro em Profundidade	0.000971	16

```
?- aStar('Sarah Silva','Pedro Mourao',3,Cam,Custo).

Tempo de geracao da solucao:0.0010042190551757812
Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Pedro Mourao'],
Custo = 16 .

?- bestfs1('Sarah Silva', 'Pedro Mourao',3,Cam,Custo).

Tempo de geracao da solucao:0.0
Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Pedro Mourao'],
Custo = 16 .

?- caminho_maisForte2('Sarah Silva', 'Pedro Mourao',3,Cam,Custo).

Tempo de geracao da solucao:0.0009708404541015625
Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Pedro Mourao'],
Custo = 16.
```

NÍVEL 4

	TEMPO	FORÇA DE LIGAÇÃO
A*	0.00	16
BestFirst	0.00	16
Primeiro em Profundidade	0.0006070	17

```
?- aStar('Sarah Silva','Pedro Mourao',4,Cam,Custo).

Tempo de geracao da solucao:0.0
Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Pedro Mourao'],
Custo = 16 .

?- bestfs1('Sarah Silva', 'Pedro Mourao',4,Cam,Custo).

Tempo de geracao da solucao:0.0
Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Pedro Mourao'],
Custo = 16 .

?- caminho_maisForte2('Sarah Silva', 'Pedro Mourao',4,Cam,Custo).

Tempo de geracao da solucao:0.0006070137023925781
Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Rafael Soares', 'Pedro Mourao'],
Custo = 17.
```

Na rede desenhada para o efeito, o BestFirst parece ser mais rápido na determinação de um caminho. O primeiro em profundidade tem a vantagem de verificar todos os caminhos e retornar o mais forte, no entanto, os recursos consumidos na procura da solução dependem do tamanho do problema. Se neste exemplo o primeiro em profundidade permite com certeza obter o melhor resultado, numa situação real poderá não ser realista a sua utilização. Num enquadramento real, por exemplo no caso de uma rede social, a melhor escolha assentaria no A*. Ainda que não garanta o caminho mais forte apresentará sempre o caminho com melhor estimativa.

IMPLEMENTAÇÃO DA FUNÇÃO MULTICRITÉRIO QUE CONTEMPLE FORÇAS DE LIGAÇÃO E DIFERENÇA ENTRE LIKES E DISLIKES

```
estimativa_multicriterio(FatorMult,NivelMax,NivelAtual,Estimativa):-
                Estimativa is FatorMult * (NivelMax-NivelAtual).

opcoes_multicriterio(Ligacao,Relacao,FatorMult):-
        (Ligacao =2,Relacao =< 0,FatorMult is 0);
        (Ligacao =2,between(1,49,Relacao),FatorMult is 25);
        (Ligacao =2,Relacao >= 50,FatorMult is 50);
        (between(3,5,Ligacao),Relacao =< 0,FatorMult is 25);
        (between(3,5,Ligacao),between(1,49,Relacao),FatorMult is 50);
        (between(3,5,Ligacao),Relacao >= 50,FatorMult is 75);
        (Ligacao =6,Relacao =< 0,FatorMult is 50);
        (Ligacao =6,between(1,49,Relacao),FatorMult is 75);
        (Ligacao =6,Relacao >= 50,FatorMult is 100).
```

De forma a limitarmos os valores apresentados pela força de relação e tomar em conta os valores da força de ligação criamos uma regra com as várias possibilidades que pode o fator multicritério apresentar (opções_multicriterio). Uma vez o fator multicritério calculado, determinamos a estimativa multiplicando o fator multicritério pela diferença entre o número de ligações máximas impostas com o nível atual.

Adaptação dos 3 métodos (Primeiro em Profundidade, Best First e A*) para considerar a função multicritério do ponto anterior

Primeiro em Profundidade

```
:-dynamic melhor_sol_maisForte_Mult/2.
:-dynamic conta_sol_maisForte_Mult/1.


caminho_maisForte2_Mult(Orig,Dest,Nivel,LCaminho_maisForte,Forca):-
        get_time(Ti),
        (melhor_caminho_maisForte2_Mult(Orig,Dest,Nivel);true),
        retract(melhor_sol_maisForte_Mult(LCaminho_maisForte,Forca)),
        retract(conta_sol_maisForte_Mult(NS1)),
        get_time(Tf),
        T is Tf-Ti,
        nl, write('Número de soluções encontradas: '),write(NS1),nl,
        nl, write('Tempo de geracao da solucao:'),write(T),nl.

melhor_caminho_maisForte2_Mult(Orig,Dest,Nivel):-
        asserta(melhor_sol_maisForte_Mult(_,0)),
        asserta(conta_sol_maisForte_Mult(0)),
        dfs(Orig,Dest,LCaminho),
        atualiza_melhor_maisForte2_Mult(LCaminho,Nivel),
        fail.

atualiza_melhor_maisForte2_Mult(LCaminho,Nivel):-
        retract(conta_sol_maisForte_Mult(NS)),
        NS1 is NS+1,
        asserta(conta_sol_maisForte_Mult(NS1)),
        melhor_sol_maisForte_Mult(_,N),
        calculateStrength2_Mult(LCaminho,[],SumStrength),
        length(LCaminho,C),
        %nl, write(LCaminho), nl,
        C1 is C-1,
        C1=<Nivel, !,
        write(SumStrength),
        nl, write("Caminho: "), write(LCaminho), nl,
        SumStrength>N,
        retract(melhor_sol_maisForte_Mult(_,_)),
        asserta(melhor_sol_maisForte_Mult(LCaminho,SumStrength)).


calculateStrength2_Mult([_],Temp,SumStrength):-
        sumList(Temp,SumStrength),!.

calculateStrength2_Mult([A,B|T], Temp, Strength):-
        ligacao(A,B,FL1,FR1),
        ligacao(B,A,FL2,FR2),
        Forca is FL1+FL2,
        Relacao is FR1+FR2,
        opcoes_multicriterio(Forca,Relacao,Soma),
        removeElement([A,B|T],A,List),
        calculateStrength2_Mult(List,[Soma|Temp],Strength).
```

BEST FIRST

```
bestfs1_Mult(Orig,Dest,NivelMax,Cam,Custo):-
        bestfs12_Mult(Dest,[[Orig]],NivelMax,0,Cam,Custo).
        %write('Caminho='),write(Cam),nl.

bestfs12_Mult(Dest,[[Dest|T]|_],_,_,Cam,Custo):-
        reverse([Dest|T],Cam),
        calcula_custo(Cam,Custo).

bestfs12_Mult(Dest,[[Dest|_]|LLA2],NivelMax,NivelAtual,Cam,Custo):-
        !,
        bestfs12_Mult(Dest,LLA2,NivelMax,NivelAtual,Cam,Custo).


bestfs12_Mult(Dest,LLA,NivelMax,NivelAtual,Cam,Custo):-
        NivelAtual=<NivelMax,
        LA=[Act|_],
        member1(LA,LLA,LLA1),

        length(LA,NumUsers),
        NivelAtual1 is NumUsers-1,

        ((Act==Dest,!,bestfs12_Mult(Dest,[LA|LLA1],NivelMax,NivelAtual,Cam,Custo));
        (NivelAtual1<NivelMax,
         findall((FatorMulti,[X|LA]),
                 (ligacao(Act,X,FLX,FRX),ligacao(X,Act,FLY,FRY),
                  Ligacao is FLX + FLY,
                  Relacao is FRX + FRY,
                  opcoes_multicriterio(Ligacao,Relacao,FatorMulti),
                   \+member(X,LA)),
                 Novos),
         Novos\==[],!,
         %nl, write('Novos= '), write(Novos),nl,
         sort(0,@>=,Novos,NovosOrd),
         %nl, write('Novos Ord= '), write(NovosOrd),nl,

         retira_custos(NovosOrd,NovosOrd1),
         append(NovosOrd1,LLA1,LLA2),
         %nl,write('LLA2='),write(LLA2),nl,
         bestfs12_Mult(Dest,LLA2,NivelMax,NivelAtual1,Cam,Custo))).
```

A*

```
aStar_Multicriterio(Orig,Dest,NivelMax,Cam,Custo):-
        forcaMaximaMulticriterio(ListaOrdenada),
        aStar3(Dest,[(_,0,0,[Orig])],NivelMax,0,ListaOrdenada,Cam,Custo).

aStar3(Dest,[(_,Custo,_,[Dest|T])|_],_,_,_,Cam,Custo):-
        reverse([Dest|T],Cam).

aStar3(Dest,[(_,Ca,Ra,LA)|Outros],NivelMax,NivelAtual,ListaFMOrdenada,Cam,Custo):-

        NivelAtual =< NivelMax,
        member(Act,LA),
        %LA=[Act|_],

        length(LA,NumUsers),
        NivelAtual1 is NumUsers-1,
        NivelAtual1 < NivelMax,
        nth0(NivelAtual1,ListaFMOrdenada,ValorEstimadoMax),

        findall((CEX,CaX,RaX,[X|LA]),
                (Dest\==Act,
                 (ligacao(Act,X,ForcaX,RelacaoX),ligacao(X,Act,ForcaY,RelacaoY)),
                 \+ member(X,LA),
                 CaX is ForcaX + ForcaY + Ca,
                 RaX is RelacaoX + RelacaoY + Ra,
                 estimativa_multicriterio(ValorEstimadoMax,NivelMax,NivelAtual1,EstX),
                 %write('Estimativa= '), write(EstX),nl,
                 CEX is EstX),
                Novos),
        append(Outros,Novos,Todos),
        write('Novos='),write(Novos),nl,
        sort(0,  @>, Todos,  TodosOrd),
        write('TodosOrd='),write(TodosOrd),nl,
        aStar3(Dest,TodosOrd,NivelMax,NivelAtual1,ListaFMOrdenada,Cam,Custo).

aStar3(Dest,[(_,_,_,_)|Outros],NivelMax,NivelAtual,ListaOrdenada,Cam,Custo):-
        aStar3(Dest,Outros,NivelMax,NivelAtual,ListaOrdenada,Cam,Custo).


forcaMaximaMulticriterio(ListaFMOrdenada):-
        findall(FactorMult,
                ((ligacao(Act,X,ForcaX,RelacaoX),ligacao(X,Act,ForcaY,RelacaoY)),
                 Forca is ForcaX+ForcaY,
                 Relacao is RelacaoX + RelacaoY,
                 opcoes_multicriterio(Forca,Relacao,FactorMult)),
                ListaTotal),
        sort(0,@>=,ListaTotal,ListaFMOrdenada).
```

COMPARAÇÃO DOS 3 MÉTODOS COM VÁRIOS EXEMPLOS E USANDO A FUNÇÃO MULTICRITÉRIO

NÍVEL 2

	TEMPO	FORÇA DE LIGAÇÃO
A*	0.02129888	6
BestFirst	0.02681207	6
Primeiro em Profundidade	0.00229811	6

```
?- aStar_Multicriterio('Sarah Silva','Pedro Mourao',2,Cam,Custo).
Cam = ['Sarah Silva', 'Rafael Soares', 'Pedro Mourao'],
Custo = 6 .

?- bestfs1_Mult('Sarah Silva', 'Pedro Mourao',2,Cam,Custo).
Cam = ['Sarah Silva', 'Rafael Soares', 'Pedro Mourao'],
Custo = 6 .

?- caminho_maisForte2_Mult('Sarah Silva', 'Pedro Mourao',2,Cam,Custo).
Cam = ['Sarah Silva', 'Rafael Soares', 'Pedro Mourao'],
```

NÍVEL 3

	TEMPO	FORÇA DE LIGAÇÃO
A*	0.02404618	6
BestFirst	0.02600383	16
Primeiro em Profundidade	0.0	16

```
?- aStar_Multicriterio('Sarah Silva','Pedro Mourao',3,Cam,Custo).
Cam = ['Sarah Silva', 'Rafael Soares', 'Pedro Mourao'],
Custo = 6 .

?- bestfs1_Mult('Sarah Silva', 'Pedro Mourao',3,Cam,Custo).
Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Pedro Mourao'],
Custo = 16 .

?- caminho_maisForte2_Mult('Sarah Silva', 'Pedro Mourao',3,Cam,Custo).
Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Pedro Mourao'],
```

NÍVEL 4

	TEMPO	FORÇA DE LIGAÇÃO
A*	0.01822710	6
BestFirst	0.02463603	16
Primeiro em Profundidade	0.00102901	16

```
?- aStar_Multicriterio('Sarah Silva','Pedro Mourao',4,Cam,Custo).
Cam = ['Sarah Silva', 'Rafael Soares', 'Pedro Mourao'],
Custo = 6 .

?- bestfs1_Mult('Sarah Silva', 'Pedro Mourao',4,Cam,Custo).
Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Pedro Mourao'],
Custo = 16 .

?- caminho_maisForte2_Mult('Sarah Silva', 'Pedro Mourao',4,Cam,Custo).
Cam = ['Sarah Silva', 'Dario Ornelas', 'Fabio Silva', 'Pedro Mourao'],
```

Como mencionado anteriormente, o uso de Primeiro em profundidade permite obter a melhor solução, no entanto, quando enfrentamos uma rede de grande tamanho este algoritmo torna-se inviável. O Best first, como métodos de pesquisa informado, apresenta uma melhor eficiência comparativamente ao anterior, mas não tão eficaz como o A*.

O A*, devido ao cálculo da estimativa, permite otimizar o resultado. No entanto, uma solução otimizada não é sinónima de melhor solução. Como exemplo, abaixo está o resultado do A* para uma rede até o 3º nível. Estando este algoritmo orientado para devolver um caminho com maior estimativa, este devolveu um caminho com uma estimativa de 200 e força de ligação igual a 6. Ora, mais à frente verifica-se a presença de um caminho com uma maior força de ligação (16) mas como possui uma estimativa mais baixa não é apresentada.

```
?- aStar_Multicriterio('Sarah Silva','Pedro Mourao',3,Cam,Custo).

TodosOrd=[(300,6,-4,[Dario Ornelas,Sarah Silva]),(300,6,-133,[Rui Mariske,Sarah Silva]),(300,2,-36,[Rafael Soar
es,Sarah Silva])]

TodosOrd=[(300,6,-133,[Rui Mariske,Sarah Silva]),(300,2,-36,[Rafael Soares,Sarah Silva]),(200,10,66,[Fabio Silv
a,Dario Ornelas,Sarah Silva]),(200,8,32,[Jonatas Granjeiro,Dario Ornelas,Sarah Silva])]

TodosOrd=[(300,2,-36,[Rafael Soares,Sarah Silva]),(200,10,66,[Fabio Silva,Dario Ornelas,Sarah Silva]),(200,9,-8
4,[Carlos Moutinho,Rui Mariske,Sarah Silva]),(200,8,32,[Jonatas Granjeiro,Dario Ornelas,Sarah Silva])]

TodosOrd=[(200,10,66,[Fabio Silva,Dario Ornelas,Sarah Silva]),(200,9,-84,[Carlos Moutinho,Rui Mariske,Sarah Sil
va]),(200,8,32,[Jonatas Granjeiro,Dario Ornelas,Sarah Silva]),(200,6,15,[Jonatas Granjeiro,Rafael Soares,Sarah
Silva]),(200,6,-36,[Pedro Mourao,Rafael Soares,Sarah Silva]),(200,5,-8,[Fabio Silva,Rafael Soares,Sarah Silva])
]

TodosOrd=[(200,9,-84,[Carlos Moutinho,Rui Mariske,Sarah Silva]),(200,8,32,[Jonatas Granjeiro,Dario Ornelas,Sara
h Silva]),(200,6,15,[Jonatas Granjeiro,Rafael Soares,Sarah Silva]),(200,6,-36,[Pedro Mourao,Rafael Soares,Sarah
 Silva]),(200,5,-8,[Fabio Silva,Rafael Soares,Sarah Silva]),(75,16,116,[Pedro Mourao,Fabio Silva,Dario Ornelas,
Sarah Silva]),(75,13,94,[Rafael Soares,Fabio Silva,Dario Ornelas,Sarah Silva])]

TodosOrd=[(200,8,32,[Jonatas Granjeiro,Dario Ornelas,Sarah Silva]),(200,6,15,[Jonatas Granjeiro,Rafael Soares,S
arah Silva]),(200,6,-36,[Pedro Mourao,Rafael Soares,Sarah Silva]),(200,5,-8,[Fabio Silva,Rafael Soares,Sarah Si
lva]),(75,16,116,[Pedro Mourao,Fabio Silva,Dario Ornelas,Sarah Silva]),(75,13,94,[Rafael Soares,Fabio Silva,Dar
io Ornelas,Sarah Silva])]

TodosOrd=[(200,6,15,[Jonatas Granjeiro,Rafael Soares,Sarah Silva]),(200,6,-36,[Pedro Mourao,Rafael Soares,Sarah
 Silva]),(200,5,-8,[Fabio Silva,Rafael Soares,Sarah Silva]),(75,16,116,[Pedro Mourao,Fabio Silva,Dario Ornelas,
Sarah Silva]),(75,13,94,[Rafael Soares,Fabio Silva,Dario Ornelas,Sarah Silva]),(75,12,83,[Rafael Soares,Jonatas
 Granjeiro,Dario Ornelas,Sarah Silva]),(75,10,76,[Pedro Mourao,Jonatas Granjeiro,Dario Ornelas,Sarah Silva])]

TodosOrd=[(200,6,-36,[Pedro Mourao,Rafael Soares,Sarah Silva]),(200,5,-8,[Fabio Silva,Rafael Soares,Sarah Silva
]),(75,16,116,[Pedro Mourao,Fabio Silva,Dario Ornelas,Sarah Silva]),(75,13,94,[Rafael Soares,Fabio Silva,Dario
Ornelas,Sarah Silva]),(75,12,83,[Rafael Soares,Jonatas Granjeiro,Dario Ornelas,Sarah Silva]),(75,10,76,[Pedro M
ourao,Jonatas Granjeiro,Dario Ornelas,Sarah Silva]),(75,8,59,[Pedro Mourao,Jonatas Granjeiro,Rafael Soares,Sara
h Silva]),(75,8,51,[Dario Ornelas,Jonatas Granjeiro,Rafael Soares,Sarah Silva])]

Tempo de geracao da solucao:0.06131100654602051
Cam = ['Sarah Silva', 'Rafael Soares', 'Pedro Mourao'],
Custo = 6
```

CONCLUSÃO

Este sprint permitiu estudar e assimilar o funcionamento de diversos métodos de pesquisa.

A pesquisa em profundidade apresenta a vantagem de ter poucos requisitos em termos de memória, no entanto não garante que a primeira solução encontrada seja a melhor solução. Este fator faz com que este seja eficiente numa amostra pequena, no entanto, numa situação real, tal como seria uma rede social, o seu uso é totalmente inviável.

O Best First é um método de pesquisa dito informado que se assemelha bastante com o primeiro em profundidade. A grande diferença situa-se no facto da decisão sobre o nó a explorar é efetuado com base numa decisão local (por exemplo a força da ligação). Este algoritmo não garante que a melhor solução seja encontrada.

O A* é o algoritmo ideal para redes de maior dimensão uma vez que permite construir uma solução com base numa estimativa. A estimativa é calculada usando a soma do custo acumulado desde o nó de origem até o nó atual e a estimativa do custo até o nó final. Ainda que não garanta a melhor solução, gera uma solução otimizada independentemente do tamanho da rede em estudo.