

Uma imagem com alimentação

Descrição gerada automaticamente

**Relatório de ALGAV**

**Turma 3DF\_ 032**

1181616 \_ Tiago Oliveira

1180604 \_ Vasco Silva

1190881 \_ Matheus Figueira

1190835 \_ Luís Teixeira

**Data: 22/01/2022**

Índice

[Parte I – Introdução e estrutura do relatório 3](#_Toc93772568)

[Parte II – Desenvolvimento 4](#_Toc93772569)

[II.1 Algoritmos desenvolvidos 4](#_Toc93772570)

[II.2 Explicação e Exemplos dos algoritmos 4](#_Toc93772571)

[II.2.1 Criação de uma rede à parte 4](#_Toc93772572)

[II:2.2 Estimativa 5](#_Toc93772573)

[II.2.3 Adaptação do A\* 5](#_Toc93772574)

[II.2.4 Adaptação do Best First 7](#_Toc93772575)

[II.2.5 Adaptação do Primeiro em Profundidade 7](#_Toc93772576)

[II.2.6 Função Multicritério 8](#_Toc93772577)

[II.2.6 Comparação dos algoritmos 9](#_Toc93772578)

[Parte III – Conclusões 12](#_Toc93772579)

# Parte I – Introdução e estrutura do relatório

Este relatório visa expor e explicar os algoritmos criados para o módulo de análise da rede social (ARS), no âmbito da Unidade Curricular de Algoritmia Avançada, lecionada (Aulas Pratico-Laboratoriais) pelo professor Jorge Coelho (JMN). Este módulo será parte integrante do projeto integrador do 5º semestre da Licenciatura de Engenharia Informática do ISEP, no ano curricular 21/22. Os algoritmos estão escritos em PROLOG.

O relatório está dividido em três partes:

* Parte 1 – Introdução e estrutura do relatório;
* Parte 2 – Desenvolvimento: onde terá a exposição dos algoritmos, respetiva implementação em PROLOG e explicação com exemplos;
* Parte 3 - Conclusão.

# Parte II – Desenvolvimento

## II.1 Algoritmos desenvolvidos

Nesta secção iremos expor os algoritmos requisitados.

* Criação de uma rede à parte – Criação de uma rede à parte com os utilizadores que podem ser alcançados até N ligações a partir de um dado utilizador;
* Estimativa – Estimativa
* Adaptação do A\* – Adaptação do A\* ao problema da determinação do caminho mais forte (máximo de N ligações);
* Adaptação do Best First – Adaptação do Best First ao problema da determinação do caminho mais forte (máximo de N ligações);
* Adaptação do Primeiro em Profundidade – Adaptação do Primeiro em Profundidade para gerar a melhor solução (já implementado no Sprint anterior) para o máximo de N ligações;
* Função Multicritério – 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.

## II.2 Explicação e Exemplos dos algoritmos

Nesta secção iremos analisar, explicar e exemplificar cada um dos casos de uso.

## II.2.1 Criação de uma rede à parte

Uma imagem com texto

Descrição gerada automaticamente

Para a implementação deste predicado foi reutilizada uma parte dos predicados desenvolvidos no sprint B, relativos ao tamanho da rede do utilizador. No cabeçalho recebemos um número de níveis N e um nó E. O predicado retorna (em L) a lista de todos os nós alcançáveis a partir do nó E, até N níveis.

Este predicado será usado, posteriormente, para o cálculo da estimativa tanto do A\* como do BFS.

## II:2.2 Estimativa

Uma imagem com texto

Descrição gerada automaticamente

Para adaptar os algoritmos de travessia (A\* e BFS) ao problema de determinação do caminho mais forte, foi desenvolvido este predicado, que calcula a estimativa do somatório das forças de ligação entre um nó e um nó destino. Como se trata de um problema de maximização, interessa que a estimativa sobrestime sempre o valor real.

O predicado começa por definir a sub-rede do nó atual (usando o predicado anterior), encontra em que nível dessa rede se encontra o nó destino (nível N), encontra o valor máximo das forças de relação e multiplica este valor por N. O resultado é essencialmente o valor máximo possível do somatório das forças entre os 2 nós.

## II.2.3 Adaptação do A\*

Para adaptar o A\* ao problema proposto basta empregar a estimativa desenvolvida e depois limitar o predicado a um número máximo de níveis.

Uma imagem com texto

Descrição gerada automaticamente

Para limitar o A\* a N níveis é importante realçar um detalhe da estimativa que não foi explicado:

Uma imagem com texto

Descrição gerada automaticamente

O valor de N (definido na estimativa) representa o número de níveis que devem ser considerados na sub-rede do nó atual. Este valor é definido como Max-N (no A\*), Max – N representa o número máximo de níveis (definido pelo utilizador) menos o comprimento do caminho atual (ou seja, o número de níveis que ainda podem ser percorridos para chegar ao destino). Se este valor for igual ou menor que 0, significa que o nó destino está fora de alcance. Sendo assim, o resultado da estimativa é definido como -9999. Isto garante que o caminho a ser considerado vai para o fundo da stack e, portanto, não é considerado para expansão.

Para demonstrar a restrição de níveis:

aStar(1,200,L,C,5).



(5 é o número de ligações. O tamanho máximo é 6)

aStar(1,200,L,C,6).

Uma imagem com texto

Descrição gerada automaticamente

O único cenário em que um caminho que excede o comprimento máximo poderia acabar no topo da “stack” (sendo, portanto considerado para expansão ou solução), seria se todos os outros caminho também excedessem o comprimento total. Isto acontece se o utilizador definir um comprimento máximo irresolúvel (menor que o shortest path). Para impedir o A\* de devolver um caminho neste cenário, é feita a verificação N<=Max (que em qualquer outra situação acaba por ser redundante). Neste cenário o A\* apenas devolve **false** .

## II.2.4 Adaptação do Best First

Uma imagem com texto

Descrição gerada automaticamente

Para adaptar o BFS foram feitas as mesmas alterações que foram feitas ao A\*.

Nota: com a estimativa desenvolvida, apesar de funcionar para o A\*, o BFS não garante que a solução seja encontrada na primeira execução. É necessária uma abordagem semelhante ao DFS (brute force).

## II.2.5 Adaptação do Primeiro em Profundidade

Uma imagem com texto

Descrição gerada automaticamente

Para adaptar o DFS foi apenas calculado o comprimento do caminho atual e feita a verificação N<=Max, filtrando os caminhos que excedem o comprimento máximo. O algoritmo funciona de forma idêntica a partir desse ponto.

## II.2.6 Função Multicritério

A função multicritério tem o objetivo de considerar tanto a força de ligação como a força de relação ao calcular uma estimativa.

Na prática só foi aplicada no A\*, mas como a estimativa funciona da mesma forma no BFS, a implementação da função multicritério seria semelhante.

Uma imagem com texto

Descrição gerada automaticamente

O predicado começa por calcular a força de ligação (FL) e relação (FR) “em percentagem” (do valor mínimo ao valor máximo).

Como a FL varia entre 0 e 100, basta dividir o valor por 100.

A FR, no entanto, varia entre -200 e 200. Então é necessário somar 200 ao valor (fazendo como se variasse entre 0 e 400), depois podemos dividir o resultado por 400 para descobrir o valor percentual.

Tendo os 2 valores percentuais, como ambos têm o mesmo peso, basta somar os 2 e multiplicar por 100. Se os 2 valores tivessem pesos diferentes, bastava multiplicar cada um pelo fator correspondente.

## II.2.6 Comparação dos algoritmos

Vamos comparar os algoritmos desenvolvidos, para tamanhos de rede diferentes.

**Para uma rede de largura 2 (1,11,12,21,22,31,32,…), comprimento máximo 6-**

DFS:

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto

Descrição gerada automaticamente

A\*:

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto

Descrição gerada automaticamente

BFS :





Neste tamanho de rede, os algoritmos DFS e BFS são consideravelmente mais rápidos.

**Para uma rede de largura 3 (1,11,12,13,21,22,23,31,32,33,…), comprimento máximo 6-**

DFS:

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto

Descrição gerada automaticamente

A\*:

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto

Descrição gerada automaticamente

BFS:





**Para uma rede de largura 4, comprimento máximo 6-**

Para este tamanho de rede o DFS torna-se irresponsivo. Demora vários minutos a chegar a um resultado e pode até atingir o limite da stack.

A\*:

Uma imagem com texto

Descrição gerada automaticamente Uma imagem com texto

Descrição gerada automaticamente

BFS:





**Conclusões**

O DFS é útil como uma solução simples para grafos pequenos, mas torna-se rapidamente insustentável. O desempenho do A\* e do BFS tornam-se mais próximos à medida que o tamanho da rede aumenta e ambos fornecem um tempo de geração razoável. No entanto, devido à abordagem “brute force” do BFS, o A\* acaba por expandir menos ramos. É, portanto, um algoritmo mais “completo” que também garante chegar à solução na primeira execução.

**Com a função multicritério-**

Como já foi explicado, a função multicritério apenas foi implementada no A\*.

Uma imagem com texto

Descrição gerada automaticamente

A função multicritério tem bastante impacto no desempenho. Este impacto é tanto maior quanto forem os valores resultantes da função (em relação aos valores reais das forças) :

Uma imagem com texto

Descrição gerada automaticamente

## Parte III – Conclusões

Para concluir, conseguimos implementar todos os algoritmos propostos para a unidade curricular de ALGAV, no entanto não conseguimos fazer a integração da função multicritério em todos. Também temos uma integração do prolog com o resto do sistema bastante limitada, o que faz com que a maioria dos dados usados nestes exemplos sejam “dummy data” que não refletem o estado atual do sistema (ex: a diferença de likes-dislikes para a função multicritério).