

**MIEIC - Concepção e Análise de Algoritmos**

**Pesquisa em Rede**

**Relatório**

**2MIEIC01 - Grupo B - Tema 4**

# Página de Rosto

**Tema do Projeto:** 2;

**Curso:** Mestrado Integrado em Engenharia Informática e Computação;

**Unidade Curricular:** Cencepção e Análise de Algoritmos.

**Ano letivo:** 2014/15;

**Grupo B:**

* João Nogueira, nº 201303882 up201303882@fe.up.pt;
* António Pedro Fraga nº 201303095 up201303095@fe.up.pt;
* Filipa Barroso nº 201307852 up201307852@fe.up.pt.

Turma 1;

**Data de Entrega:** 1 de junho de 2015

**Índice**

[Página de Rosto 2](#_Toc420893189)

[Introdução 4](#_Toc420893190)

[Explicação do Problema 5](#_Toc420893191)

[Descrição do Problema 6](#_Toc420893192)

[Input 6](#_Toc420893193)

[Introdução de dados 6](#_Toc420893194)

[Output 6](#_Toc420893195)

[Objetivo 6](#_Toc420893196)

[Formalização do Problema 7](#_Toc420893197)

[Input 7](#_Toc420893198)

[Output 7](#_Toc420893199)

[Solução 8](#_Toc420893200)

[Algoritmos 8](#_Toc420893201)

[Esquema do programa 9](#_Toc420893202)

[Casos de Utilização 10](#_Toc420893204)

[Métricas de avaliação 10](#_Toc420893205)

[Avaliação empírica do seu desempenho 10](#_Toc420893206)

[Complexidade temporal 10](#_Toc420893207)

[Diagrama de Classes 11](#_Toc420893208)

[Principais dificuldades 12](#_Toc420893209)

[Esforço por elemento 12](#_Toc420893210)

[Código 12](#_Toc420893211)

[Introdução de Dados 12](#_Toc420893212)

[Implementação Gráfica 12](#_Toc420893213)

[Implementação de algoritmos 12](#_Toc420893214)

[Relatório 12](#_Toc420893215)

[Conclusão 13](#_Toc420893216)

# Introdução

Foi no âmbito da Unidade Curricular de Concepção e Análise de Algoritmos do 2º ano do MIEIC que nos foi apresentado um problema que nos foi pedido para resolver da forma mais eficiente possível.

Neste relatório está explicado o problema em si, por forma a que se entenda exatamente qual o nosso objetivo e como o pretendemos atingir.

Depois de explicitar qual o problema que procuraremos resolver, explicamos neste relatório os métodos que usaremos para a resolução do problema em questão.

# Explicação do Problema

As redes sociais são uma realidade cada vez mais presente e indispensável na socialização para além da interação pessoal. Tendo em conta que nas redes sociais uma pessoa tem um número de amigos maior que o essencial e também há a tendência de procurar por pessoas conhecidas recentemente, daí a necessidade de uma ferramenta de pesquisa de pessoas.

Na ferramenta que desenvolvemos, cabe ao utilizador introduzir o nome da pessoa que está à procura. Esta ordena os resultados pela semelhança com o nome introduzido e pela “proximidade” com o utilizador, ou seja, se determinada pessoa for amiga do utilizador terá maior probabilidade de correspondência com este do que uma pessoa com o mesmo nome mas que não é amiga do utilizador.

.

# Descrição do Problema

De uma forma geral, pretendemos otimizar a pesquisa de perfis de uma determinada rede social.

## Input

Construção de um grafo, G = (V, E), de nomes de perfis e amizades no qual:

* V – vértices – representam todos os nomes incluídos numa rede social;
* E – arestas – representam todas as amizades entre perfis de uma rede social.
* Nome de origem de pesquisa e nome a pesquisar.

## Introdução de dados

Dois ficheiros, um deles com todos os nomes existentes e outro com pares de id’s, que representam amizades. Para criação de vários dados inicias para teste criámos um programa auxiliar que submetemos juntamente com o projeto.

## Output

Lista de nomes ordenados pelos critérios usados, sendo estes (pela mesma ordem de aplicação):

1. Semelhança entre strings.
2. Número de níveis em profundidade no grafo.

## Objetivo

Possibilitar aos utilizadores uma pesquisa com maior possibilidade de sucesso.

# Formalização do Problema

Formalizamos agora o problema, de acordo com aquela que achamos ser a melhor forma para resolver aquilo a que nos propusemos.

## Input

## Output

# Solução

## Algoritmos

Pensamos que o melhor algoritmo a ser usado neste trabalho é a **pesquisa em Largura,** que expande e examina sistematicamente todos os vértices de um grafo direcionado ou não-direcionado. Por outras palavras, podemos dizer que o algoritmo realiza uma busca exaustiva a um grafo passando por todas as arestas e vértices do grafo.

O objetivo é assim descobrir todos os vértices com uma distância n ao vértice da raiz antes de todos os vértices com uma distância maior. Essa característica do algoritmo permite construir uma árvore de distâncias mínimas (menor número de arestas) entre o vértice raiz e os demais

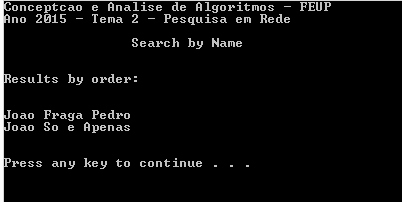
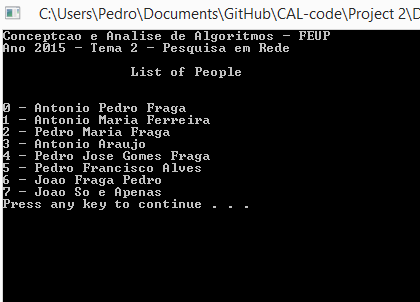
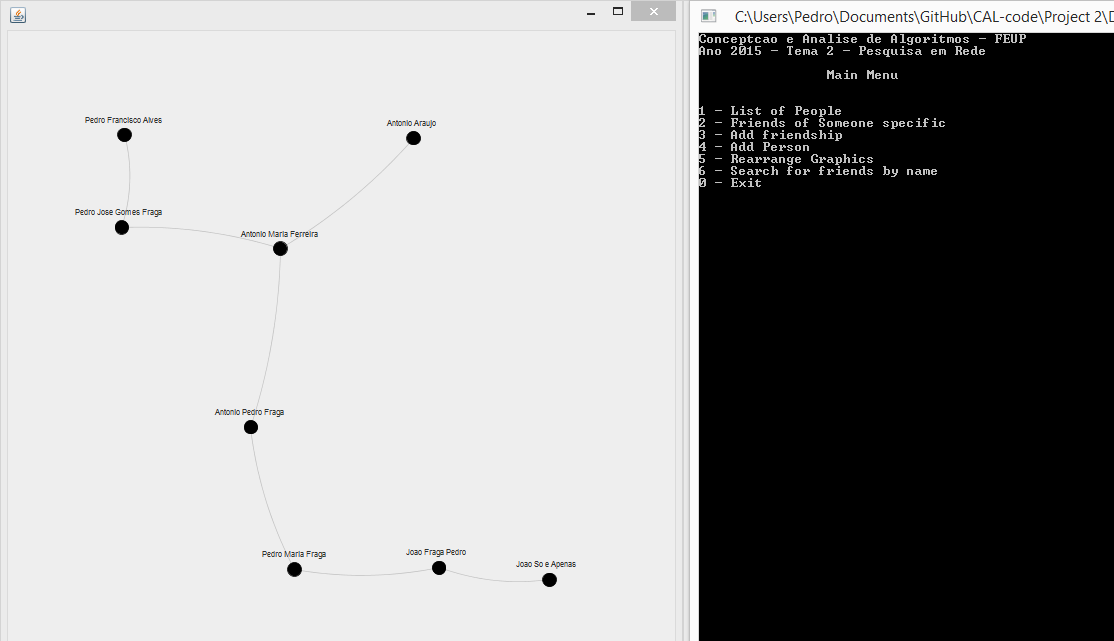
Para pesquisa de strings, aplicamos o **algoritmo de Knuth–Morris–Pratt**, que procura uma ocorrência de uma string dentro de uma substring empregando a simples técnica de que quando aparece uma diferença, a palavra tem em si a informação necessária para determinar onde começar a próxima comparação.

## Esquema do programa

Começamos por introduzir dados de graphos, vértices (perfis) e arestas (amizades). De seguida, agrupamos os graphos em dois tipos de estruturas de dados. Uma das estruturas é utilizada para representação gráfica e outra delas é utilizada para a pesquisa, pesquisa essa efetuada pelo algoritmo descrito em cima. As duas estruturas de dados apresentam ligeiras diferenças, de forma a tornar a representação gráfica mais “*user friendly*”.

Depois de carregar todos os dados, o ultilizador pode escolher listas todos os perfis, adicionar amizades ou pessoas, ver todos os amigos da pessoa ‘X’ ou reorganizar os grafos na sua representação gráfica. Quando quiser efetuar a sua pesquisa, introduz qual o perfil a partir do qual quer pesquisar e o nome de pesquisa. Ser-lhe-ão apresentados os resultados da sua pesquisa em forma de lista.

# 



# Casos de Utilização

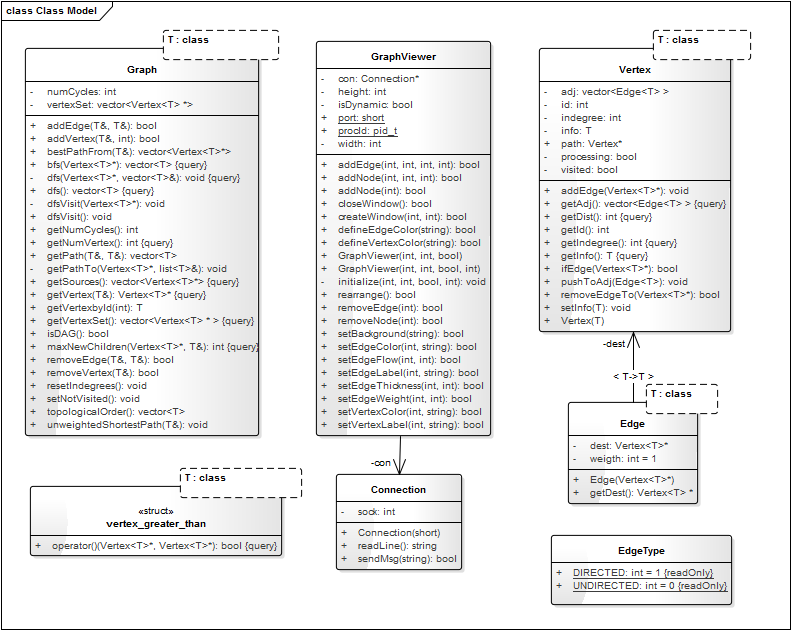
* Listagem de todos os perfis;
* Listagem de todos os amigos de determinada pessoa;
* Pesquisa a partir de um perfil de um determinado nome, ou conjunto de nomes;
* Visualização da rede de perfis através do GraphViewer.

# Métricas de avaliação

## Avaliação empírica do seu desempenho

Para avaliar e testar a complexidade temporal dos algoritmos utilizamos diferentes dados de entrada.

# Diagrama de Classes



# Esforço por elemento

## Código

### Introdução de Dados

No que diz respeito à procura e forma de introdução dos dados da rede de transportes o João Nogueira foi quem supervisionou. Porém contudo, esta foi feita também pelos outros elementos.

### Implementação Gráfica

Em relação à implementação da parte Gráfica, todos os elementos participaram de igual forma e ativamente, tendo esta sido feita paralelamente à implementação do algoritmo.

### Implementação de algoritmos

Em relação à implementação do algoritmo, todos participaram na implementação estando o Pedro Fraga responsável por esta componente.

## Relatório

No que diz respeito ao relatório, todos os elementos do grupo se esforçaram igualmente para a sua realização.

# Conclusão

Toda a nossa rede social foi preenchida com dados fictícios. Em primeiro lugar foram introduzidos manualmente nomes de pessoas. Para criar as amizades criámos um programa para associar os perfis aos pares. Isto fêz-nos entender que muitas vezes será necessário desenvolver outras ferramentas, que por muito simples que sejam, são necessárias ao funcionamento do nosso programa.

A realização deste trabalho serviu para ficarmos mais inteirados da matéria em questão, particularmente com o modo de funcionamento da pesquisa em largura e do algoritmo de Knuth–Morris–Pratt, e percerbermos como funciona uma pesquisa em rede, provavelmente um método utilizado nas redes sociais atuais.