

**Mestrado Integrado em Engenharia Informática e Computação**

**Conceção e Análise de Algoritmos**

**Central de atendimento de urgências**

Turma 4 / Grupo 9

Carlos Miguel da Silva de Freitas (201504749) - [up201504749@fe.up.pt](mailto:up201504749@fe.up.pt)

Luís Noites Martins (201503344) – [up201503344@fe.up.pt](mailto:up201503344@fe.up.pt)

Rui Emanuel Cabral de Almeida Quaresma (201503005) - [up201503005@fe.up.pt](mailto:up201503005@fe.up.pt)

9 abril 2017

Índice

[Introdução 4](#_Toc479351914)

[Identificação do Problema 5](#_Toc479351915)

[Descrição do Problema 6](#_Toc479351916)

[Input 6](#_Toc479351917)

[Output 6](#_Toc479351918)

[Dados de entrada 7](#_Toc479351919)

[Restrição 7](#_Toc479351920)

[Objetivo 7](#_Toc479351921)

[Formalização do Problema 8](#_Toc479351922)

[Input 8](#_Toc479351923)

[Output 8](#_Toc479351924)

[Objetivo 8](#_Toc479351925)

[Restrição 8](#_Toc479351926)

[Solução implementada 9](#_Toc479351927)

[Algoritmos 9](#_Toc479351928)

[Funcionamento do programa 11](#_Toc479351929)

[Análise da complexidade 14](#_Toc479351930)

[Avaliação analítica da complexidade temporal 14](#_Toc479351931)

[Avaliação analítica da complexidade espacial 14](#_Toc479351932)

[Avaliação empírica da complexidade temporal 15](#_Toc479351933)

[Lista de casos de utilização 16](#_Toc479351934)

[Diagrama de classes 17](#_Toc479351935)

[Principais dificuldades 18](#_Toc479351936)

[Soluções encontradas 18](#_Toc479351937)

[Contribuição dos membros do grupo 19](#_Toc479351938)

[Conclusão 20](#_Toc479351939)

Introdução

Este trabalho foi realizado no âmbito da Unidade Curricular de Conceção e Análise de Algoritmos do 2º ano do MIEIC.

O objetivo é recorrer aos algoritmos de cálculo de semelhança de strings, abordados nas aulas, para encontrar uma solução para um tema que nos foi proposto.

Neste relatório é descrito o problema, feita a sua formalização, bem como apresentada a solução implementada. São ainda elencadas as principais dificuldades sentidas na elaboração do trabalho, bem como as soluções encontradas para as ultrapassar.

No final, apresenta-se uma breve conclusão, que inclui uma apreciação sobre a aprendizagem decorrente deste trabalho.

Identificação do Problema

Numa situação de emergência em que há necessidade de verificar de imediato quais os veículos que se encontram numa determinada localização, existe uma dificuldade acrescida, pelo facto de, com frequência, ocorrerem erros na introdução dos dados da localização pelo utilizador.

Perante este problema, pretende-se obter uma solução que permita, em tempo útil, e mediante os dados fornecidos pelo utilizador, determinar com precisão qual a localização em causa.

Descrição do Problema

Pretende-se identificar os veículos existentes num determinado local escolhido pelo utilizador. Para tal, este escreve o nome da freguesia e da rua pretendida (string), informação que é posteriormente lida e analisada.

Input

Grafo, G = (V, A), de cruzamentos e ruas em que:

* V (vértices): representam pontos de interceção de duas ruas, ou seja, cruzamentos;
* A (arestas): ligações entre dois cruzamentos (distâncias entre dois vértices);

O utilizador terá que inserir uma string, com a qual se deve verificar a semelhança com as freguesias existentes.

Terá também de introduzir uma nova string, mas desta vez para identificar a rua que pretende.

Output

Ao longo do programa vão sendo listadas as freguesias e ruas disponíveis, para que o utilizador possa saber quais pode escolher.

No fim da execução serão apresentados os veículos e respetivos nós onde se encontram, considerando a freguesia e rua escolhidas.

Dados de entrada

* Seis ficheiros que contêm os nós (cruzamentos) do mapa, com os ids, coordenadas x e y dos mesmos:
* Um ficheiro que contém os nós onde não está localizado nenhum veículo ou hospital. Contém os ids, o x e o y dos nós;
* Um ficheiro que contém os nós onde estão localizadas as ambulâncias;
* Um ficheiro que contém os nós onde estão localizados os carros dos bombeiros;
* Um ficheiro que contém os nós onde estão localizados os carros da PSP;
* Um ficheiro que contém os nós onde estão localizados os hospitais.
* Um ficheiro que contém as freguesias, com os respetivos ids, nomes e ids das ruas e nós que cada uma contém.
* Um ficheiro que contém as arestas, ou seja, as ligações entre os diferentes vértices, com os ids das mesmas e os ids dos nós de início e de fim de cada uma.
* Um ficheiro que contém as ruas, ou seja, conjuntos de arestas que podem ser unidirecionais ou bidirecionais, com os ids e o nome das mesmas, a indicação sobre se são ou não bidirecionais e ainda os ids dos nós que as compõem.

Objetivo

Providenciar ao utilizador informação sobre quais os veículos de emergência existentes na rua solicitada, tendo em conta a freguesia escolhida.

Formalização do Problema

Input

Output

Objetivo

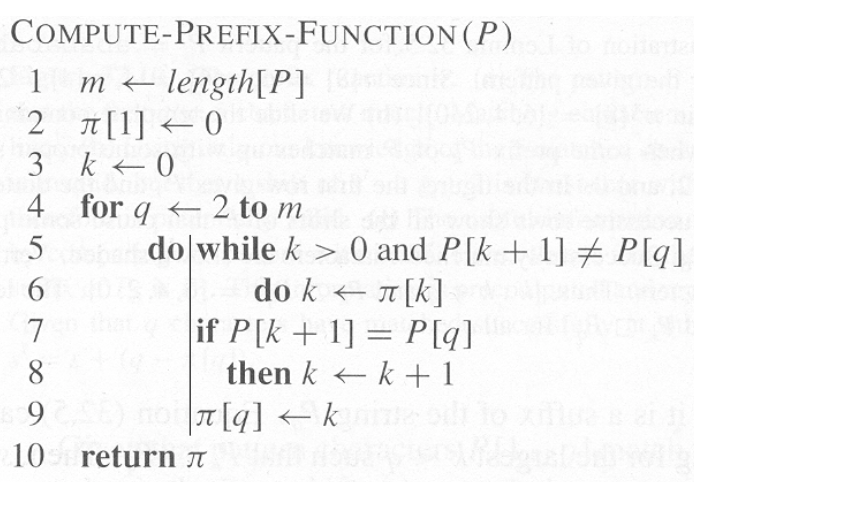
Solução implementada

Algoritmos

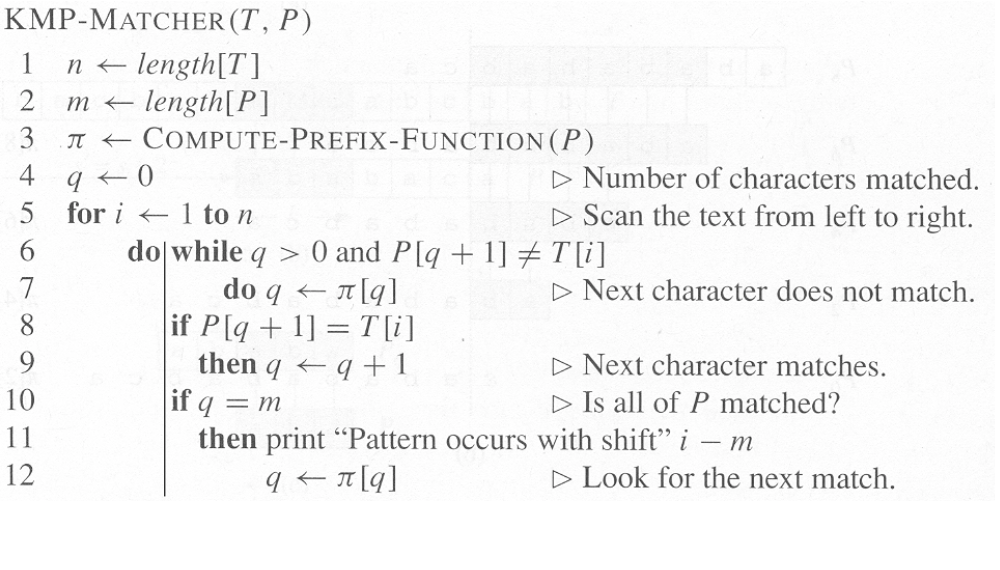
Foram implementados dois algoritmos de pesquisa em strings, um de pesquisa exata e outro aproximada, tanto para a pesquisa pelo nome das freguesias como pelo nome das ruas. Para a pesquisa exata, foi utilizado o algoritmo de Knuth-Morris-Pratt (KMP).

O algoritmo de Knuth-Morris-Pratt tem como objetivo verificar se uma dada string (padrão) está contida numa outra string (texto).

O KMP começa por realizar um pré-processamento do padrão, que serve para que o mesmo carater seja verificado mais do que uma vez. Este cria um array auxiliar com tamanho igual a |padrão| que irá conter em cada posição um prefixo para cada posição do padrão (determina se os prefixos do padrão são substrings deles mesmos).

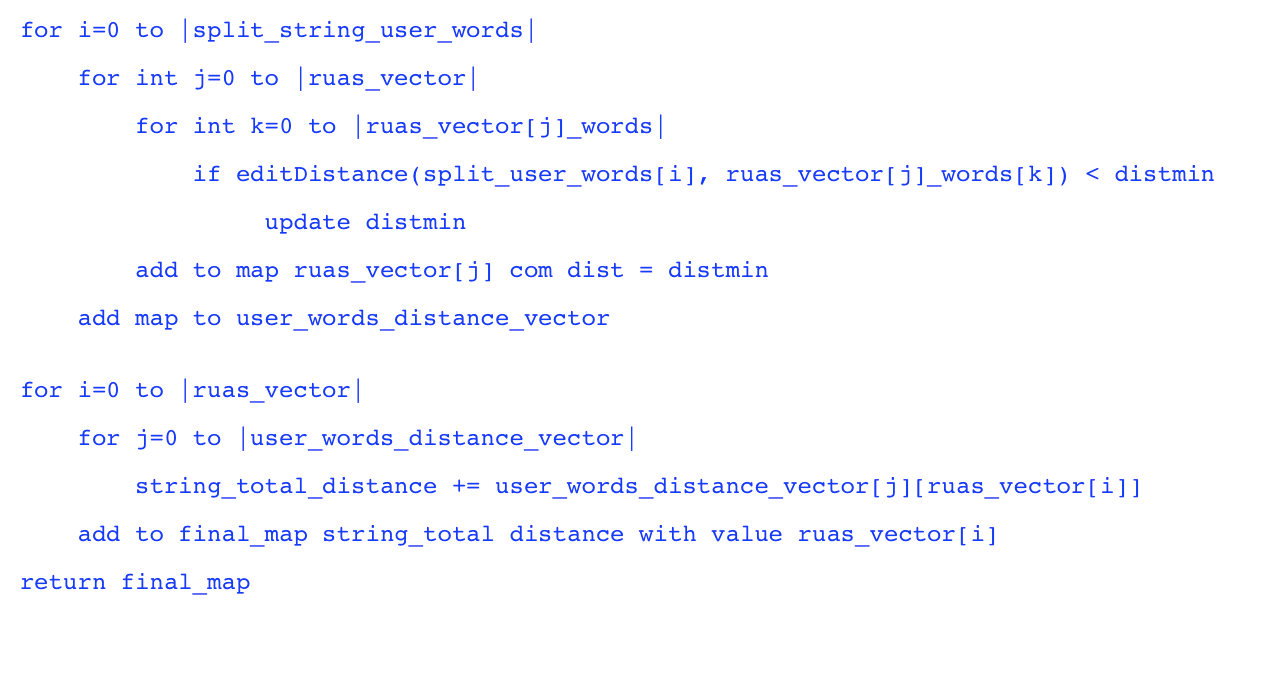


De seguida, o algoritmo começa a percorrer a string texto e a compará-la com a string padrão. Quando os carateres não são iguais, a pesquisa inicia-se no carater seguinte, não sendo necessário voltar a comparar os carateres que já tenham sido igualados.



Para a pesquisa aproximada foi utilizado o algoritmo lecionado nas aulas teóricas para descobrir a distância entre 2 strings. Uma vez que o nome de cada rua tem várias palavras e o utilizador também pode introduzir mais do que uma palavra, recorreu-se então à separação de strings antes da verificação da distância entre elas.

Assim, para cada palavra introduzida pelo utilizador, percorre-se todas as ruas existentes. Cada rua é separada nas diferentes palavras que a constituem. Para cada uma destas, calcula-se a distância entre esta e a palavra do utilizador que está a ser analisada. Se esta distância for inferior à distância mínima atribuída na relação entre a palavra do utilizador e a rua em análise, então esta distância mínima é atualizada. Desta maneira preenche–se um vetor em que a cada posição corresponde uma das palavras introduzidas pelo utilizador e, para cada palavra existe um multimap, no qual a chave é a distância mínima da rua à palavra do utilizador e o valor é o nome da rua.

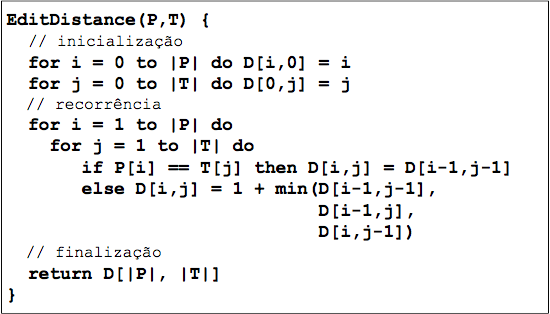


Para calcular a distância entre cada par de palavras (sendo a string proveniente da rua considerada como string padrão e a string do utilizador a string a ser testada) foi então utilizado o algoritmo da pesquisa aproximada, no qual é preenchida uma matriz D de largura igual ao tamanho da string a analisar (T) e altura igual ao tamanho da string padrão (P).

Esta análise é feita através da iteração da matriz e da análise dos valores já conhecidos. Assim, a primeira linha e primeira coluna são a distância de cada uma das palavras à string vazia e como tal estes valores irão de 0 até ao valor do comprimento de cada uma das strings.

Posteriormente, o algoritmo itera a matriz, começando em 1 (pois a linha e a coluna 0 já estão preenchidas), até ao tamanho de cada uma das strings. Para cada valor da matriz compara-se cada par de valores de P e T e avalia-se se estes são iguais ou não. Se forem, então é porque não existe nenhuma alteração de padrão e, portanto, o valor da matriz naquela posição é igual à que se encontra na posição diagonal desta mesma, na linha e coluna anteriores, ou seja, mantém o seu valor de distância. Se, por outro lado, estes forem diferentes, então o valor da matriz naquela posição é igual ao mínimo de entre as 3 posições que rodeiam a posição em consideração (à esquerda, em cima, e na diagonal) mais um, uma vez que é necessário considerar esta nova alteração.

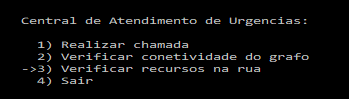
Desta forma, no final do preenchimento da matriz, o valor que se encontra em D[P.length][T.length] corresponde à distância mínima entre as 2 strings.



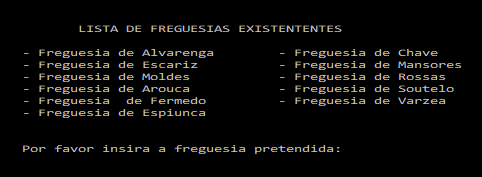
Funcionamento do programa

Para implementar a solução para o problema em causa, foram utilizados os algoritmos acima referidos.

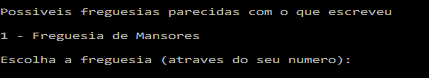
Assim, para além do funcionamento já referido no relatório do primeiro trabalho, agora o utilizador tem também a opção de verificar quais os veículos existentes numa rua através da escolha da freguesia e nome da rua.



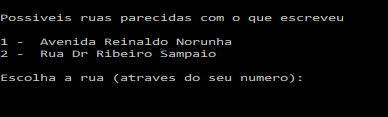
Após a seleção da opção de verificação de ruas, o utilizador poderá selecionar a freguesia que pretende, escrevendo o nome da mesma.



Após este ponto, executa-se uma pesquisa da string escrita, primeiro por verificação exata e, caso esta não exista, por verificação aproximada, retornando os valores obtidos. Neste momento, caso existam várias opções, o utilizador selecionará a pretendida pelo seu número na listagem.



Após escolher a freguesia aparecerá uma lista com o nome das ruas existentes nesta freguesia, e o utilizador poderá neste momento introduzir o nome da rua que pretende. Serão novamente executados os algoritmos de pesquisa em strings e retornados os valores obtidos.



Após isto são mostrados no ecrã os veículos disponíveis nos nós da rua selecionada que pertencem à freguesia que o utilizador selecionou.

FALTA IMAGEM

Análise da complexidade

Avaliação analítica da complexidade temporal

A complexidade temporal do algoritmo de pesquisa aproximada é , tal como se pode verificar pela explicação encontrada na solução implementada

Já no caso do algoritmo KMP, a sua complexidade temporal do algoritmo é . Neste caso em particular, existe uma complexidade

.

Avaliação analítica da complexidade espacial

A complexidade espacial do algoritmo de Knuth-Morris-Pratt é

Já para o algoritmo de verificação de strings aproximadas a complexidade espcial é

Avaliação empírica da complexidade temporal

Para analisar a complexidade temporal, medimos os tempos de execução do programa, usando diferentes dados de entrada, fazendo variar o número e tamanho das strings para uma mesma rua e freguesia.

Obtivemos desta forma os seguintes resultados:

Lista de casos de utilização

* Leitura dos dados de ficheiros representativos das componentes de um mapa;
* Identificação dos veículos existentes numa determinada rua de uma freguesia
* Visualização do mapa usando o GraphViewer;

Principais dificuldades

No início do projeto, deparamo-nos com alguns entraves à sua execução, nomeadamente a dificuldade em saber qual a melhor maneira para que o algoritmo de semelhança de string por pesquisa aproximada retornasse os valores pretendidos.

Soluções encontradas

Como referido anteriormente, foram feitas algumas divisões das strings em palavras, para que as comparações fossem mais credíveis e assim se obtivessem resultados mais semelhantes ao pretendido.

Contribuição dos membros do grupo

Todos os membros contribuíram para a realização do projeto. A distribuição do trabalho foi a seguinte:

Carlos Freitas:

* Desenvolvimento da interface do utilizador (menus e display dos resultados).

Luís Martins:

* Implementação do algoritmo de pesquisa aproximada, incluindo a divisão de strings e possíveis condições de paragem
* Redação do relatório

Rui Quaresma:

* Criação e leitura dos ficheiros utilizados
* Implementação do algoritmo KMP
* Participação na redação do relatório

Conclusão

Perante o problema apresentado no início deste relatório, pretendia-se determinar quais os veículos disponíveis no local escolhido pelo utilizador, sendo esta escolha feita através da introdução de strings.

Foi feita a análise do input do utilizador com recurso a 2 algoritmos distintos, tendo sido obtidos resultados satisfatórios.

Conclui-se que através da verificação da diferença entre strings se consegue facilmente, e em tempo útil, obter resultados semelhantes aos pretendidos pelo utilizador. Verificou-se ainda que a análise por pesquisa exata é mais rápida. No entanto, esta nem sempre surte efeito, pelo que a pesquisa aproximada é um recurso fundamental para este tipo de casos.

O tema deste trabalho representa uma situação que, embora aqui seja analisada de forma simplificada, corresponde a uma situação real, pois a pesquisa e os resultados obtidos são algo importante para que se realize uma pesquisa rápida e correta mesmo que existam erros de input.