

Relatório do Trabalho Prático 2 - Air Shuttle

Grupo E:
Gonçalo Ribeiro
up201403877
goncalo.ribeiro@fe.up.pt

29/05/2016

Introdução

Na continuação do primeiro trabalho prático, foi-nos proposto a implementação de funcionalidades de pesquisa com *strings*, no trabalho prático previamente desenvolvido. O problema central é fazer *queries*, exatas ou aproximadas, a nomes de clientes ou ruas, de forma eficiente, minimizando os tempos de pesquisa.

Em relação ao trabalho anterior, agora é possível obter informação relativa aos clientes, filtrada pelo próprio nome. Para além disso, também é possível pesquisar clientes pelo nome da rua do seu destino.

Especificações

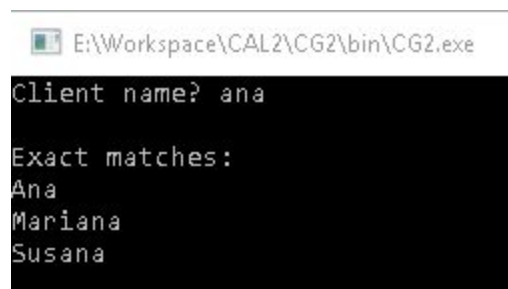
Algoritmos implementados

Pesquisa Exata

Para a pesquisa exata, decidi implementar o algoritmo **Knuth-Morris-Pratt (KMP)**.

A minha implementação recebe como parametros dois strings, S e K, e retorna um vetor dos indices dos substrings em K iguais a S.

Na pesquisa exata de clientes por nome ou morada, invoca-se o algoritmo KMP para, simplesmente, verificar a existência do string-padrão no nome em questão. Originalmente, usar-se-ia os vetor de índices para fazer highlight aos substrings iguais ao string-padrão. Acabei por não implementar essa funcionalidade, o que torna “redundante” parte do algoritmo KMP.



```
E:\Workspace\CAL2\CG2\bin\CG2.exe
Client name? ana
Exact matches:
Ana
Mariana
Susana
```

Exemplo de uma pesquisa exata pelo nome de cliente. São retornados todos os clientes cujo nome contenha o string-padrão.

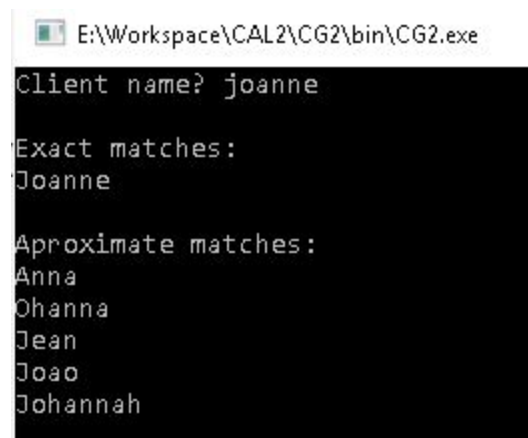
Pesquisa Aproximada

Para pesquisa aproximada, decidi implementar o algoritmo de calculo de **distância de Levenshtein**.

A minha implementação recebe como parametros dois *strings*, e retorna um inteiro correspondente à “distância” entre ambos. O conceito de distância, no que diz respeito a *strings*, representa o menor número de modificações (substituições, inserções e remoções de caracteres) que permitem transformar um *string* noutro.

Na pesquisa de clientes por nome, o algoritmo é invocado para cada entrada na lista de clientes. Apenas os clientes cujo nome dista menos que um determinado “limite aceitável” são selecionados pela pesquisa. Este “limite aceitável” varia **logaritmicamente** com o tamanho da *string*-padrão, de forma a ser mais **permissivel para padrões pequenos** e mais **restrito para padrões grandes**.

Na pesquisa por aderesso, foi necessário tomar em consideração a pluralidade de palavras contidas na morada. Para tornar a pesquisa mais eficaz a encontrar resultados, divide-se o nome das moradas em tokens. Posteriormente, calcula-se a distância entre os tokens e o *string*-padrão. Para cada morada, toma-se em consideração a **menor distância token-padrão** para deliberar os resultados mais aproximados.



```
E:\Workspace\CAL2\CG2\bin\CG2.exe
Client name? joanne

Exact matches:
Joanne

Approximate matches:
Anna
Ohanna
Jean
Joao
Johannah
```

Exemplo de uma pesquisa aproximada pelo nome de cliente.

Conclusão

Dificuldades

A maior dificuldade sentida neste trabalho foi calcular para cada caso específico, que “distância” entre *string* é considerada aceitável para considerar dois *strings* como aproximados.

Inicialmente, o limite de distância aceitável aumentava linearmente com o tamanho do *string*-padrão. Para padrões pequenos, era uma solução aceitável, mas quanto maior fosse o padrão, maior era a permissibilidade de distância e mais imprecisos eram os resultados.

Para ultrapassar o problema, teria que arranjar um limite permissível o suficiente para padrões pequenos e restrito para padrões grandes. Acabei por adoptar o limite supracitado, que varia logaritmicamente com o tamanho do *string*-padrão.

Contribuição

Gonçalo Ribeiro - 100%