**Faculdade de Engenharia da Universidade do Porto**



**Relatório Projeto CAL**

Parte 2

*RideSharing: partilha de viagens*

20 maio 2018

Grupo 10 – Turma 2

Gonçalo Santos 201603265 up201603265@fe.up.pt

João Vieira 201603190 up201603190@fe.up.pt

Susana Lima 201603634 up201603634@fe.up.pt

Conteúdo

[Introdução 2](#_Toc514466321)

[Explicação do problema 3](#_Toc514466322)

[Descrição do problema 4](#_Toc514466323)

[Dados de Entrada 4](#_Toc514466324)

[Pesquisa Exata e Aproximada 4](#_Toc514466325)

[Dados de Saida 4](#_Toc514466326)

[Pesquisa Exata 4](#_Toc514466327)

[Pesquisa Aproximada 5](#_Toc514466328)

[Objetivo 5](#_Toc514466329)

[Descrição da solução 6](#_Toc514466330)

[Estruturas de dados utilizadas 6](#_Toc514466331)

[Representação de um grafo adequado ao problema 6](#_Toc514466332)

[Gestão de passageiros e condutores 6](#_Toc514466333)

[Algoritmos implementados 7](#_Toc514466334)

[Análise da solução -TODO 8](#_Toc514466335)

[Casos de utilização 9](#_Toc514466336)

[Esforço dedicado por cada elemento do grupo -TODO 10](#_Toc514466337)

[Conclusão-TODO 11](#_Toc514466338)

[Referências Bibliográficas 12](#_Toc514466339)

[Apêndice A: Pseudocódigo do Algoritmo – TODO 13](#_Toc514466340)

# Introdução

No âmbito da unidade curricular de Conceção e Análise de Algoritmos do 2º ano do MIEIC foi proposta a criação de uma aplicação que gerencie a partilha de viagens (*RideSharing*) entre passageiros e condutores e algoritmos que a sustentem da forma mais eficiente possível usando os conhecimentos adquiridos nas aulas.

Neste relatório são descritos o problema e a melhor estratégia encontrada para a sua resolução.

# Explicação do problema

Sendo o conceito de ride sharing cada vez mais presente e existindo uma panóplia de destinos tão variada que seria impossível tornar cada um deles um ponto de interesse na nossa aplicação, existe uma necessidade de implementarmos a pesquisa de destinos, especificamente ruas – Edges - tornando mais fácil saber se o destino real (podendo não ser o ponto de interesse selecionado – Vertex) do passageiro se encontra no percurso do condutor.

Comodamente, também implementamos a pesquisa por pessoas específicas na viagem especificada.

# Descrição do problema

Existe pretensão de otimizar a pesquisa, tanto de nomes de ruas como de passageiros, o mais possível.

## Dados de Entrada[[1]](#footnote-2)

### Pesquisa Exata e Aproximada

Passageiros

* Informação do Driver em questão:
  + name – nome do condutor
  + age – idade do condutor
  + passengersPickedAt – multimap para pesquisa com nomes dos passageiros
* patern – substring a encontrar
* maximumEditDistance – distância de edição máxima entre duas strings – apenas para pesquisa aproximada

Ruas

* Informação do Driver em questão:
  + name – nome do condutor
  + age – idade do condutor
  + streets – list para pesquisa com nomes das ruas
* patern – substring a encontrar
* maximumEditDistance – distância de edição máxima entre duas strings – apenas para pesquisa aproximada

## Dados de Saida

### Pesquisa Exata

Passageiros e Ruas

* Names – vector de strings com nomes onde foi encontrada a substring patern
* Found – informação sobre a existência ou não do condutor

### Pesquisa Aproximada

Passageiros e Ruas

* patternAndDistance <name, dist> :
  + name – string onde foi encontrada a substring patern
  + dist - distância de edição entre name e patern

## Objetivo

A solução ótima encontrada passa por minimizar o tempo de pesquisa necessário.

# Descrição da solução

## Estruturas de dados utilizadas

### Representação de um grafo adequado ao problema

#### Vertex

A classe *Vertex* representa um vértice de um grafo. No contexto apresentado representa uma via de ligação entre dois pontos do mapa.

#### Edge

#### Graph

### Gestão de passageiros e condutores

#### Person

A classe *Person* representa, como o nome indica, uma pessoa no contexto do problema.

#### Driver

A classe *Driver* representa um condutor. Esta classe deriva da classe Person, uma vez que um condutor é uma pessoa sendo que possui os atributos que a caracterizam. Cada condutor, além das estruturas referidas no relatório passado, contém:

* *streets* – list de strings com os nomes das ruas incluídas no percurso do condutor;

#### Passenger

A classe *Passenger* representa um passageiro no contexto apresentado. Esta classe deriva da classe *Person* uma vez que que um passageiro é uma pessoa pelo que possui os mesmos atributos

#### RideShare

A classe *RideShare* é a entidade que permite fazer a gestão dos condutores e dos passageiros.

Algoritmos implementados

Tendo, no cálculo do caminho, já sidos selecionados tantos os passageiros da viagem como o percurso a ser realizado e sendo que estes já estão guardados em estruturas de dados especificas para o efeito, é realizada uma pesquisa sequencial por estas estruturas.

Reconhecemos que, visto que os objetos guardados nesta estrutura (list das ruas) são strings, poderíamos, de alguma forma, otimizar esta pesquisa, tratando de ordenar a mesma, podendo assim mudar o algoritmo de pesquisa utilizado. Por outro lado, mantendo a ordem pela qual foram adicionados os dados, preservamos um registo do itinerário, facilitando a mostragem do mesmo. (não sei se concordo com isto, o algoritmo ia ser o mesmo independentemente de estar ordenado)

Para a pesquisa em strings, implementamos os algoritmos de Knuth-Morris-Pratt e o da Distância de edição entre duas strings. O primeiro algoritmo referido é mais eficiente que o algoritmo naive, pois quando ocorre uma incompatibilidade entre as strings em questão, este consegue determinar onde começar a próxima comparação. Já o segundo algoritmo permite também a procura de strings em strings, neste caso admitindo um determinado grau de erro (previamente definido. Acho que isto não se devia dizer uma vez que o algoritmo em si não obriga a especificar uma distancia maxima, foi uma coisa que nos acrescentamos depois porque faz mais sentido nos menus)

# Análise da solução -TODO

Seja P o padrão a procurar e T o texto onde procurar, sendo, por conseguinte, |P| e |T| os tamanhos respetivos das duas strings.

Algoritmo de Distância de edição entre duas strings

(acho que este textinho chega não? Não se esta muito parecido com o do ferrolho :/ Falta o do outro algoritmo)

A distância de edição corresponde ao número de alterações necessárias para converter P em T. Estas podem ser:

* substituição de carateres
* inserção de carateres
* eliminação de carateres

Foi implementada uma versão melhorada do algoritmo de Distância de edição entre duas strings de modo a permitir a otimização de espaço. Em vez de se utilizar uma matriz bidimensional, usa-se um vetor de tamanho |T| em conjunto com duas variáveis que guardam os valores necessários ao preenchimento do vetor.

A complexidade temporal deste algoritmo será, portanto O(|P|.|T|), uma vez que o algoritmo analisa a totalidade de carateres de P e de T.

A complexidade espacial, em consequência da otimização implementada é O(|T|).

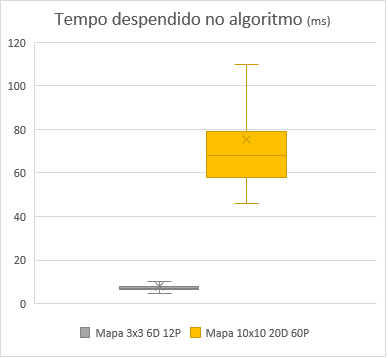
 No pior caso, a complexidade do algoritmo descrito é de O(D\* (E + V + P)logV ). A complexidade espacial é O(D + E + V + P), uma vez que são usadas referências para os objetos ao longo do algoritmo.

Figura 1 - Tempo despendido no algoritmo em função do mapa, número de passageiros e condutores

A solução apresentada não é ótima, uma vez que não é capaz de, por exemplo, deixar de apanhar um grupo que não lhe encha o carro para mais à frente, apanhar outro grupo que preencha a totalidade do carro. Não sendo ótima, apresenta uma complexidade temporal bastante aceitável. Num caso prático, a solução seria implementada a nível local e não é esperado que existam pontos de coleta em cada rua/esquina, pelo que o número de vértices não seria muito grande.

O algoritmo pode ser caracterizado por apresentar caminhos flexíveis, uma vez que o condutor não faz necessariamente o seu caminho mais curto da origem ao destino, podem haver desvios em função do número de passageiros, *multi-hop*, isto é, um passageiro pode fazer parte do seu percurso com um condutor e o resto/outra parte com outro e suporta múltiplos passageiros, isto é, mais do que um grupo de passageiros por veículo.

# Casos de utilização

A plataforma implementada permite simular o processo de gestão de um mapa e geração de percursos de *ride sharing* para diferentes condutores e passageiros. Em particular, para além das funcionalidades já referidas no relatório anterior, o programa apresenta as seguintes opções e funcionalidades:

* Listagem do itinerário do condutor, por nomes das ruas;
* Procura de passageiros por nome, no veículo de um condutor específico;
* Procura de ruas por nome, no itinerário de um condutor específico;

Esforço dedicado por cada elemento do grupo -TODO

A realização do projeto em questão foi possível devido ao empenho e esforço conjunto de todos os elementos do grupo, de uma forma organizada e sincronizada de modo a que não houvesse grande disparidade de contribuição entre os diferentes elementos. Cada elemento focou-se mais no módulo com que tinha mais à vontade, mas no geral o trabalho foi divido equitativamente entre todos. (podia ficar igual não?)

Conclusão-TODO

O programa implementado permite resolver, de uma forma geral, o problema apresentado, cumprindo o propósito do trabalho. O algoritmo concebido mostrou-se eficiente, apresentando sempre tempos e resultados satisfatórios, face aos dados fornecidos. (mudo conforme a análise da solução)

A realização deste trabalho permitiu, não só aprofundar os conhecimentos adquiridos nas aulas teóricas e práticas, como também mostrar a aplicabilidade de certos algoritmos apresentados na unidade curricular de CAL na resolução de problemas atuais e presentes na vida quotidiana de muitas pessoas.

# Referências Bibliográficas

Masoud, Neda, and R. Jayakrishnan. 2017. "A real-time algorithm to solve the peer-to-peer ride-matching problem in a flexible ridesharing system." *Transportation Research Part B: Methodological* 106: 218 - 236.

Faria, João Pascoal, Rossetti, Rosaldo, Ferreira, Liliana. Algoritmos de Pesquisa em Strings.

(acho que é assim)

# Apêndice A: Pseudocódigo do Algoritmo – TODO

(Será necessário viste que e mesmo chapar aquilo que vem nos ppt?)

(acho que não é preciso)

1. Tendo em conta que esta pesquisa é feita em torno de estruturas já descritas no relatório anterior. [↑](#footnote-ref-2)