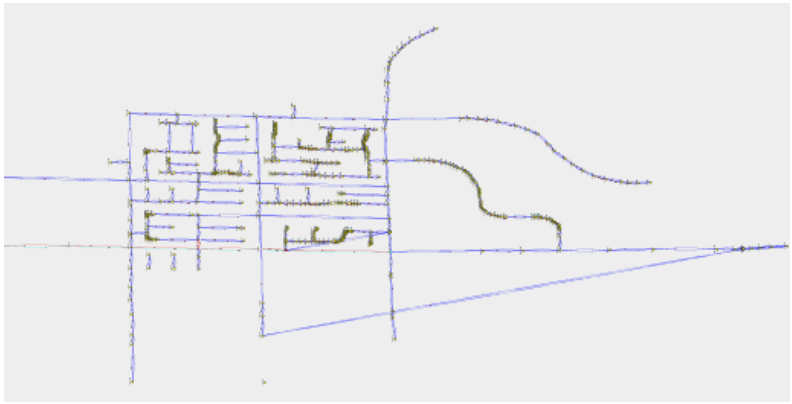
U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Supermercado ao Domicílio - Final

Mestrado Integrado em Engenharia Informática e Computação

Concepção e Análise de Algoritmos



Turma 2, grupo A:

Cristiana Maria Monteiro Ribeiro - *up201305188@fe.up.pt*

Marta Diogo Torgal Pinto - *up201407727@fe.up.pt*

Mónica Ariana Ribeiro Fernandes - *up201404789@fe.up.pt*

07/04/2017

Índice

CAL

Introdução

Este projeto foi desenvolvido na Unidade Curricular Concepção e Análise de Algoritmos do Mestrado Integrado em Engenharia Informática e Computação da Faculdade de Engenharia da Universidade do Porto.

Neste relatório encontra-se o tema escolhido pelo nosso grupo, qual a abordagem escolhida para alcançar a solução, os resultados esperados e as principais dificuldades durante a elaboração do mesmo.

Análise do problema

O tema escolhido foi o Supermercado ao Domicílio (tema 6) cujo principal objetivo é rentabilizar o serviço de entrega das encomendas feitas pelos clientes, sendo que o fator decisivo na escolha dos itinerários é a distância.

Na primeira fase do projeto, consideramos apenas a existência de um só ponto de distribuição (um só Supermercado) a partir do qual serão calculadas as distâncias mínimas para os pontos de entrega, tendo sempre como objetivo minimizar o número de viagens e maximizar o número de entregas feitas na mesma viagem.

Deste modo, as encomendas serão agrupadas pela data da sua entrega e é com base nesse critério que os camiões serão carregados. Considerando a possibilidade do peso da mercadoria exceder o limite de peso suportado pelo camião, tornou-se necessário investir em mais do que um camião para o transporte da mercadoria na mesma data.

Na segunda fase da elaboração do projeto, será acrescentada uma nova funcionalidade: pesquisa de *strings*, cujo principal objetivo é verificar se na rua introduzida pelo utilizador existe ou não algum supermercado. Para isso, consideramos a existência de uma cadeia de supermercados com vários pontos de distribuição espalhados pelo mapa.

É expectável que o utilizador introduza o nome (ou parte dele) da rua que pretende procurar no sistema. A partir daí, o programa realiza a pesquisa exata com a string introduzida e caso a pesquisa seja bem-sucedida, verifica se existe algum supermercado nessa rua. No caso desta pesquisa falhar, o programa efectua a pesquisa aproximada que indicará ao utilizador os nomes de ruas mais semelhantes, ou seja, em que a distância entre a *string* introduzida e o nome da rua for menor. De seguida, o utilizador pode escolher qual a rua que pretende analisar e o programa verifica a existência ou não de algum supermercado nessa mesma rua.

Durante a execução do programa, são mostradas mensagens informativas na consola para que o utilizador perceba qual das pesquisas está a ser efetuada.

A elaboração do projeto baseou-se na estrutura de dados grafo que foi construído com

recurso a um mapa real. O mapa real usado foi obtido através do site *https://www.openstreetmap.org/*. Assim sendo, os nós do grafo representam tanto os supermercados como as casas que vão ser alvo da entrega e as arestas representam os itinerários escolhidos desde o supermercado até aos postos de entrega.

Com recurso à aplicação *GraphViewer* conseguimos ilustrar as rotas das viagens e os nomes das ruas, bem como o resultado das pesquisas, uma vez que se estas forem bem-sucedidas e o supermercado existir, o seu ícon representativo será diferenciado.

Formalização do tema

Dados de entrada

Para ler e guardar a informação necessária à execução do programa, utilizámos alguns ficheiros de texto:

- *roads.txt* – guarda o identificador e o nome da estrada, bem como um indicador de bidirecionalidade (isto é, se a rua é de um ou de dois sentidos).
- *users.txt* – guarda o nome dos clientes do supermercado, o seu número de identificação fiscal e o identificador da sua casa (nó) .
- *edges.txt* – guarda o identificador da estrada a que a aresta pertence, bem como o seu nó de origem e o nó de destino.
- *vertex.txt* – guarda o identificador do nó, a sua latitude e longitude em graus e em radianos.
- delivery.txt - guarda a informação das entregas a realizar a casa, o peso da encomenda e a data a realizar a entrega.

Os ficheiros correspondentes aos nós, às arestas e às estradas foram obtidos com recurso ao programa cedido (*Parser*) na página do moodle da unidade curricular de modo a obter mapas reais usámos o OpenStreetMaps(*https://www.openstreetmap.org/*).

A partir destes construímos um grafo pesado ***G =< V, E >*** em que:

- ➔ ***V*** - Vértice ou nó que representa as conexões entre as arestas. Alguns deles representam o supermercado e as casas às quais se pretende realizar as entregas.
- ➔ ***E*** - Aresta representa uma rua (ou parte dela - menos comum). O seu peso é dado pela distância entre o nó de origem e o nó de destino.

Para a implementação da pesquisa de *strings*, é pedido ao utilizador que introduza a *string* que deverá ser analisada pelo programa.

Dados de saída

O resultado do programa, isto é, as casas nas quais foram feitas entregas e o percurso percorrido pelo camião, é apresentado na forma de gráfico com o suporte da ferramenta *Graph Viewer*, como foi referido anteriormente.

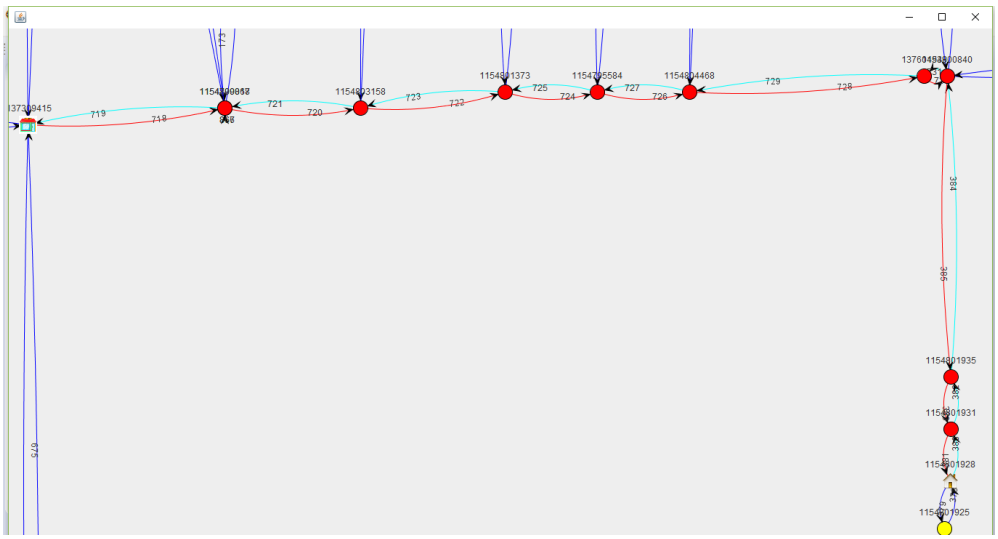


Figura 1: Mapa de entregas.

Quando é solicitada a pesquisa de uma *string*, para além das mensagens informativas (que explicitam o sucesso ou insucesso da pesquisa), no caso de existir um supermercado na rua procurada, o programa altera o seu *ícon* correspondente para o utilizador poder ver com clareza a sua localização no mapa.

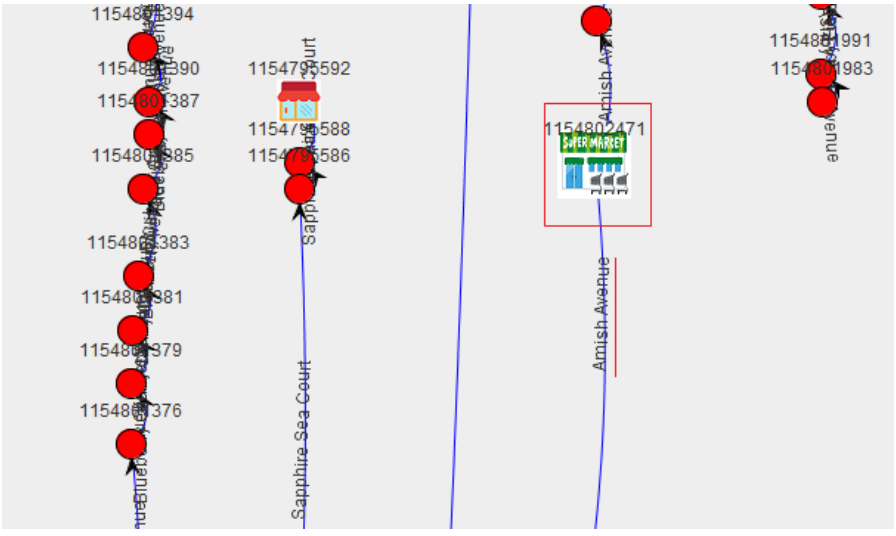


Figura 2: Supermercado resultante da pesquisa.

Função Objetivo

CAL

- Efetuar o maior número de entregas possível (desde o supermercado até ao destino) no mesmo dia, escolhendo sempre o caminho mais curto, nunca esquecendo a necessidade de respeitar a capacidade máxima dos camiões.
- Tendo sido fornecido o nome de uma rua (ou parte dele), verificar se a rua existe e, em caso afirmativo, indicar se existe ou não um supermercado nessa mesma rua.

Funcionalidades da aplicação

A aplicação começa por mostrar um menu, em que o utilizador pode escolher quais são as atividades que pretende realizar:

```
Bem-vindo ao Super Precos Baixos!!
1 - Clientes
2 - Distribuicao de entregas
3 - Administrar Supermercado
4 - Pesquisar Ruas
5 - Sair.
Insira o numero da sua escolha. Obrigado.
```

Clientes

Registar-se

```
Bem-vindo a tab dos Clientes!
1 - Registar-se
2 - Login
3 - Sair.
Insira o número da sua escolha. Obrigado.
1
Indique por favor:
O seu nome
Marta Torgal
O seu nif:
123456789
O id da sua morada:
154025730
Ja se encontra registado!
```

Login

```
Bem-vindo a tab do Login!
Introduza o seu nif
123456789
Bem-vindo a tab dos Clientes, 123456789!
1 - Realizar uma encomenda
2 - Modificar Conta
3- Eliminar Conta
4 - Ver perfil
5- Sair
```

1. **Realizar uma encomenda**

```
Indique o peso da sua encomenda!
50
Indique a data de entrega da encomenda!
09/04/2017
```

2. **Modificar conta**

```
Modificar conta de Cliente
1 - Modificar o Nome da conta
2 - Modificar a Morada
3 - Sair
```

3. **Eliminar conta**

A conta do utilizador que fez o Login é eliminada.

Distribuição de entregas

```
id Truck 27
 truck date: 12/12/2017;;;
 truck capacity max: 500
 truck max distance: 1.93622e+006

Indique, por favor, o id do camiao!
```

Após o utilizador escolher um id (do camião) da lista que lhe é mostrada, o utilizador pode visualizar qual é o itinerário feito por esse camião (imagem na secção Dados de Saída).

Administração do supermercado

```
Bem-vindo ao tab de administracao do Super Preços Baixos
1 - Mudar o Nome do Supermercado
2 - Modificar o local do Supermercado
3 - Modificar Camioes
4 - Sair.
Insira o número da sua escolha. Obrigado.
```

1. Mudar nome do Supermercado

```
Insira o número da sua escolha. Obrigado.
1
Indique o novo nome do Supermercado
Compraki
Bem-vindo ao tab de administracao do Compraki
```

2. Modificar o local do Supermercado

```
2
Indique a nova morada(id) do Supermercado
137319915
A morada não se encontra no mapa!
```

3. Modificar Camiões

```
Insira o número da sua escolha. Obrigado.
3
1 - Modificar a capacidade maxima do camiao
2 - Modificar a distancia maxima percorrida pelo camiao
3 - Sair.
Insira o número da sua escolha. Obrigado.
1
```

Pesquisa de Ruas

Pesquisa Exata

```
Insira o numero da sua escolha. Obrigado.
4
Bem-Vindo à pesquisa de Supermercado por rua! Pf insira o nome de uma rua que deseja procurar.
Beel
Inicio de pesquisa exata...
A rua que procurava foi encontrada: Beeline Court
Supermercado nao encontrado na rua pesquisada.
Bem-vindo ao Super Precos Baixos!!
1 - Clientes
2 - Distribuicao de entregas
3 - Administrar Supermercado
4 - Pesquisar Ruas
5 - Sair.
Insira o numero da sua escolha. Obrigado.
```

Pesquisa Aproximada

```
4
Bem-Vindo à pesquisa de Supermercado por rua! Pf insira o nome de uma rua que deseja procurar.
Chs
Inicio de pesquisa aproximada...
A rua que procurava foi encontrada: Chris Craft Street
Supermercado nao encontrado na rua pesquisada.
```

Pesquisa com sucesso

```
Insira o numero da sua escolha. Obrigado.
4
Bem-Vindo à pesquisa de Supermercado por rua! Pf insira o nome de uma rua que deseja procurar.
mish
Inicio de pesquisa exata...
A rua que procurava foi encontrada: Amish Avenue
Supermercado encontrado na rua pesquisada.
```

Limites da Aplicação

1. Os limites da aplicação recaem sobre a capacidade máxima que o camião pode transportar, assim como a data em que a entrega se realiza. Datas diferentes, implicam viagens diferentes e a sobrecarga de um camião também. O camião também tem um limite máximo de distância que pode percorrer numa viagem, e se este for ultrapassado, então, não é possível efetuar a entrega.
2. Como utilizámos uma pequena parte do mapa, nem todos os nós estão ligados e é possível que existam casas em que não se podem efetuar entregas.
3. Para descobrir a distância real entre dois vértices foi necessário recorrer a fórmulas matemáticas.
   3.1. De forma a converter a longitude e a latitude em coordenadas planas aplicámos as seguintes fórmulas:

   x *= Raio_da_Terra * cos (latitude) * cos (longitude);*
   y *= Raio_da_Terra * cos (latitude) * sin (longitude);*

   3.2. Para calcular a distância entre dois vértices:

CAL

***deltaLat** = latitude1 - latitude2;*

***deltaLon** = longitude 1 - longitude2;*

***c** = 2* asin (√a);*

***a** = sin(deltaLat/2)* sin(deltaLat/2) +*
*sin(deltaLon/2)*sin(deltaLon/2)*cos(latitude1)*cos(latidude2);*

***Distância =** 100* c* Raio_da_Terra ;*

Resultados Esperados

- Espera-se que as despesas da companhia de supermercados sejam reduzidas, dado que o número de entregas durante uma única viagem será maximizado.
- Também é expectável que as ruas pesquisadas pelo utilizador sejam encontradas (caso existam) e que os supermercados sejam diferenciados.

Será possível visualizar o percurso de cada viagem e o resuldado da pesquisa com a ajuda da ferramenta *GraphicViewer.*

Descrição da solução

Técnicas de Concepção

Das diversas classes do programa destacam-se as seguintes:

- Company - Classe responsável por interligar o grafo, o supermercado e o graphviewer.
- Supermarket - Classe do supermercado. Guarda os camiões, e todos os dados do supermercado, bem como os utilizadores.
- Graph - Responsável por criar o grafo (inclui as classes Vertex e Edge).
- Info - Classe de informação. Esta classe representa o tipo de cada Vertex.
- Graphviewer - Responsável por criar a animação do grafo.

- Truck - Classe do camião. Guarda as suas características (data da entrega, distância máxima, capacidade máxima, distância percorrida, capacidade utlizada) assim como as suas encomendas.
- User - Responsável por guardar os dados dos utilizadores da aplicação.
- Order - Classe da encomenda. Guarda as suas características (data, peso, etc).
- Road – Classe responsável por guardar as informações das ruas presentos no programa.

Algoritmos

1. Dijkstra

Optámos por usar este algoritmo, uma vez que é capaz de calcular o menor caminho entre dois nós de um grafo, e que os pesos, neste projeto, não são negativos.

Grande parte do algoritmo tinha sido implementada nas aulas práticas, contudo, algumas alterações foram acrescentadas no sentido de o adaptar ao tema que nos foi proposto.

Este método tem uma complexidade temporal de $O(|E|*log\ |V|)$, em que "E" é o número de arestas e "V" o número de vértices.

A complexidade espacial é de ordem $O(1)$.

2. Travel Salesman Problem

Travel Salesman é um algoritmo para determinar o menor caminho a percorrer vários pontos/cidades (visitando apenas uma única vez cada uma deles), retornando ao ponto/cidade de origem. Este problema de otimização é inspirado na necessidade dos vendedores ao realizar entregas em diversos locais percorrendo o menor caminho possível, reduzindo o tempo necessário para a viagem e os possíveis custos de transporte.

CAL

Este problema foi também o problema do nosso projecto. Tentamos recolher informação sobre este algoritmo de modo a melhorar a abordagem ao nosso tema.

3. Knuth-Morris-Pratt

Este algoritmo foi implementado para realizar a pesquisa exata e a sua função é procurar a ocorrência de uma *string* num texto.

Knuth-Morris-Pratt possui complexidade temporal e espacial de O(|T| + |P|), em que |T| representa o tamanho da string de texto e |P| representa o tamanho da string que é procurada nesse texto.

.

4. Pesquisa Aproximada

O algoritmo de Pesquisa Aproximada verifica quais as *strings* de texto que necessitam de menor número de alterações relativamente a uma palavra que queiramos procurar e retorna-as.

Como medimos o tempo de execução desta pesquisa, comprava-se que quanto maior for o número de alterações a efetuar numa palavra, maior é o tempo de execução do algoritmo, comprovando a dependência deste em relação ao tamanho da *string* de texto e de padrão.

Este algoritmo possui complexidade temporal e espacial de O(|P|.|T|).

Diagrama de Classes

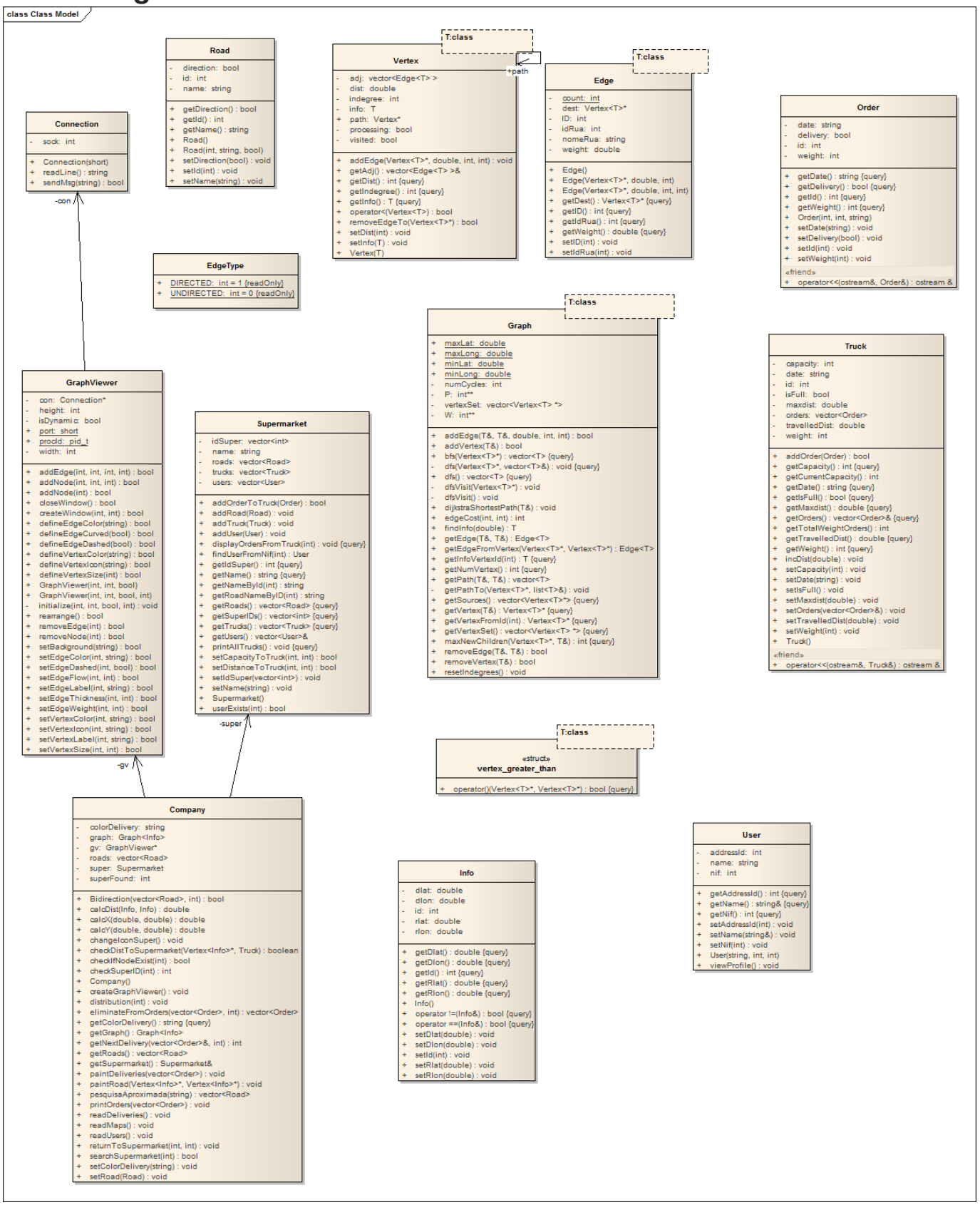


Figura 3: Diagrama de classes UML.

Foi anexado ao projecto uma pasta com a imagem do diagrama de classes UML, para melhor visualização e realização de zoom se necessário.

Conclusão

O desenvolvimento deste projeto permitiu a todos os elementos do grupo compreender e utilizar uma estrutura de dados tão importante como é o grafo. Permitiu também o uso de programação dinâmica, nos seus diversos tipos.

Com a análise e pesquisa de quais seriam os melhores algoritmos a implementar na nossa aplicação, aprendemos um pouco mais no que diz respeito a esse aspecto. Aliás, encontrar o algoritmo que melhor se aplicava ao problema, foi uma das dificuldades principais com que o grupo se deparou.

Agora que o projeto está finalizado, achámos que fomos capazes de encontrar a melhor solução para a resolução do problema e acreditamos que cumprimos todos os requisitos.

O trabalho foi distribuído de igual forma por todos os elementos, por isso estas são as percentagens que atribuímos a cada um :

Cristiana Maria Monteiro Ribeiro - *33.33%*

Marta Diogo Torgal Pinto - *33.33%*

Mónica Ariana Ribeiro Fernandes - *33.33%*

Bibliografia

- *http://www.movable-type.co.uk/scripts/latlong.html*
- *https://en.wikipedia.org/wiki/Travelling_salesman_problem*
- Página da Unidade Curricular do Moodle.