

# Concepção e Análise de Algoritmos

## Distribuição de telecomunicações

Turma 2 - Grupo C

José Peixoto	200603103	ei12134@fe.up.pt
Pedro Moura	201306843	up201306843@fe.up.pt

30 de Março de 2015

## Conteúdo

<b>1</b>	<b>Formalização do problema</b>	<b>3</b>
1.1	Dados de entrada . . . . .	3
1.2	Limites e condições de aplicação . . . . .	3
1.3	Resultados esperados . . . . .	4
<b>2</b>	<b>Perspectivas de solução</b>	<b>4</b>
<b>3</b>	<b>Métricas de avaliação</b>	<b>5</b>
<b>A</b>	<b>Map</b>	<b>6</b>

## Resumo

No âmbito da unidade curricular de Concepção e Análise de Algoritmos foi-nos proposto o desenvolvimento de uma aplicação para planificação da distribuição de fibra óptica numa aldeia que evidencie o conhecimento e uso adequado de algoritmos sobre grafos. Reúnem-se dados sobre coordenadas geográficas e distâncias através do sítio da Internet *OpenStreetmap*. Após a construção de um grafo que represente a área em estudo, são-lhe aplicados algoritmos que determinam a melhor solução para o problema na minimização da distância e dos custos da instalação de fibra óptica.

# 1 Formalização do problema

## 1.1 Dados de entrada

Mapeia-se uma área, usando coordenadas geográficas que representem intersecções entre ruas ou coordenadas de casas. Para este efeito, usam-se dados extraídos do OpenStreetMap<sup>1</sup> sobre a área seleccionada.

Construção de um grafo a partir de vértices-casa, cruzamento ou central  $V$  e arestas  $E$  que representam as distâncias entre as casas ligadas pelas ruas.

$$G < V, E > \quad (1)$$

Pretende-se guardar os dados em dois ficheiros distintos, **vertexes.csv**, onde temos os números identificativos das casas e intersecções de ruas, vértices do grafo e **edges.csv** onde se estabelecem as relações entre os vértices do ficheiro anterior e respectivas distâncias, por forma a caracterizar as arestas do grafo.

Os vértices são guardados um por cada linha e são separados por ponto e vírgula de um valor que determina se o vértice é intersecção de rua -  $0$ , casa -  $1$  ou central -  $2$ .

Exemplo:  $3;1$  (o vértice número 3 é uma casa)

As arestas são guardadas uma por cada linha, sendo os primeiros dois valores os números identificativos dos vértices e o último a distância em metros entre eles.

Exemplo:  $2;3;150$  (o vértice número 3 dista 150 metros do vértice 2)

## 1.2 Limites e condições de aplicação

Na lista final, não poderá existir repetição de pares de vértices, ou seja, arestas repetidas e todos os vértices-casa têm de estar contidos nesta lista final.

---

<sup>1</sup><https://www.openstreetmap.org/#map=17/41.83596/-8.42833>

Caso os dados sejam recolhidos de forma manual, torna-se difícil a representação de uma área grande, com muitos vértices dos quais resulte um grafo de maiores dimensões, pela morosidade de todo o processo de escrita nos ficheiros atrás mencionados. Os resultados da aplicação também estarão dependentes da qualidade e precisão dos valores iniciais, nomeadamente das coordenadas e das distâncias. Em consequência dos algoritmos usados, referidos na secção seguinte, os grafos alvo terão de ser obrigatoriamente não dirigidos e conexos.

### 1.3 Resultados esperados

A aplicação determinará as coordenadas dos vértices que definem as arestas seleccionadas para instalação da cablagem com fibra óptica e a distância total mínima, soma da distância destas arestas. O subconjunto das arestas que formam a árvore de expansão mínima serão armazenadas num ficheiro *output.csv*, com formatação idêntica à do ficheiro das arestas.

$$Output = List < V_i, V_j > V_i, V_j \in V_s \quad (2)$$

A função objectivo retornará a distância mínima  $D$  do somatório das distâncias parciais  $dist$  das arestas formadas pelos pares de vértices escolhidos.

$$D = \min(\sum_{i,j} dist(V_i, V_j)): V_i, V_j \in Output_k \quad (3)$$

## 2 Perspectivas de solução

Após análise do enunciado, verificou-se que um método adequado na resolução do problema será a utilização de um algoritmo que determine uma árvore de expansão mínima, já que se pretende passar fibra por todas as casas de forma o mais eficiente possível, ou seja, com menor gasto de cabo fibra óptica, reduzindo os custos para a empresa. Temos em mente a utilização de 3 algoritmos:

- Algoritmos que determinam árvores de expansão mínima:

**Prim** é um algoritmo ganancioso que encontra a árvore de expansão mínima para um grafo não direccionado, ou seja, este algoritmo tenta encontrar um subconjunto das arestas que formam uma árvore contendo todos os vértices onde o peso total de todas as arestas é mínimo. É mais eficiente do que o algoritmo de *Kruskal* se o número de arestas for muito elevado em relação aos vértices.

**Kruskal** tal como o algoritmo de *Prim* serve para encontrar a árvore de expansão mínima. É mais adequado do que o algoritmo *Prim* para grafos mais simples com poucas arestas, pois usa estruturas mais simples.

- Algoritmos que determinam o caminho mais curto:

**Dijkstra** com programação dinâmica é usado para encontrar os caminhos mais curtos entre os vértices de um grafo. Neste caso será necessário para procurar o caminho mais curto entre casas que tenham uma ou mais intersecções de ruas entre elas, ou seja, como uma intersecção de ruas pode não ser um vértice casa, pois não contém uma casa, selecciona-se o caminho mais curto entre dois vértices-casa para que este se torne como que uma aresta entre essas duas casas de forma a ser usado correctamente no algoritmo usado para determinar a árvore expansão de custo mínimo.

### 3 Métricas de avaliação

Como se vai trabalhar com grafos de pequena dimensão, no teste da aplicação, julga-se necessária a criação de testes unitários que executem de forma repetida o mesmo código numa ordem de magnitude o suficientemente grande para se efectuarem medições com valores de múltiplos de segundo.

Para a medição da complexidade espacial, ter-se-á em conta a memória necessária para armazenar todas as variáveis criadas na execução do algoritmo e crescimento com o aumento da quantidade dos dados de entrada.

---

N.B. Este relatório não foi escrito ao abrigo do novo Acordo Ortográfico.

## A Map

