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

26 de Abril de 2015

Conteúdo

1 Formalização do problema **3**
1.1 Dados de entrada 3
1.2 Limites e condições de aplicação 4
1.3 Situações de contorno 4
1.4 Resultados esperados 4

2 Descrição da solução **4**
2.1 Algoritmo de Kruskal 5

3 Lista de casos de utilização **6**

4 Relato das principais dificuldades encontradas no desenvolvimento do trabalho **6**

5 Indicação do esforço dedicado por cada elemento do grupo **7**
5.1 José Peixoto 7
5.2 Pedro Moura 7

A Map **8**

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[1] 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.

$$G < V, E > \tag{1}$$

Área máxima A em m^2, do círculo com centro na central da companhia Telefones, resultando na exclusão das casas fora dessa selecção.

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`, a partir do qual se estabelecem as relações entre os vértices do ficheiro anterior e respectivas distâncias, por forma a caracterizar as arestas do grafo.

É lido um vértice por cada linha, contendo o *id* e são separados por ponto e vírgula dos valores posicionais x e y e de um valor que determina se o vértice é uma intersecção de ruas - 0 ou uma casa - 1.

Exemplo: `3;264;250;1` (o vértice número 3 é uma casa com coordenadas x e y de 264 e 250 pixeis respectivamente)

É lida uma aresta 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] Mapa exemplo em anexo

1.2 Limites e condições de aplicação

A lista final não pode conter pares de vértices, ou seja, arestas repetidas e todos os vértices-casa têm de estar presentes nesta lista.

Como as coordenadas e distâncias exemplo são escritos em ficheiros separados por vírgulas de forma manual, torna-se morosa a representação de uma área muito grande, com muitos vértices e arestas, dos quais resulta num grafo de maiores dimensões. Os resultados da aplicação também estão dependentes da qualidade, precisão e realismo dos valores das coordenadas e das distâncias.

Como requisito do algoritmo usado, referido na secção seguinte, os grafos alvo terão de ser obrigatoriamente não dirigidos, conexos e pesados.

1.3 Situações de contorno

No mapeamento dos vértices, por forma a serem adequadamente enquadrados na resolução escolhida pelo visualizador gráfico de grafos, convencionou-se que a área total é limitada a 480000 pixeis^2 (800×600). Para manter as proporções mais ou menos coerentes, estabeleceu-se que 1 pixel é equivalente a 2 metros. Apesar de o programa ler a área em metros quadrados, é feita internamente a conversão para unidades pixel, usando o rácio atrás referido.

Para reduzir a complexidade do código, o programa decide se tem de gerar posições para as centrais caso nos ficheiros dos vértices não encontre nenhum desse tipo.

1.4 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 resultante é acíclico, formando uma árvore de expansão mínima.

$$Output = Set < V_i, V_j > V_i, V_j \in V_s \tag{2}$$

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

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

2 Descrição da solução

Após análise cuidada do enunciado, concluiu-se que um método adequado na resolução do problema passa pela utilização de um algoritmo que determine a árvore de expansão mínima, uma vez que se pretende instalar fibra óptica

em todas as casas de forma o mais eficiente possível, ou seja, com menor gasto de cabo de fibra óptica, reduzindo os custos para a empresa.

O grafo alvo é construído a partir da leitura de dois ficheiros: um com vértices e outro com arestas. São feitas verificações da correcção dos valores *id*, coordenadas x e y e o tipo de intersecção. De igual modo, na leitura das arestas e em cada linha, são verificados os valores identificadores do vértice fonte e do vértice destino e a sua distância. Para garantir que se trata de um grafo não dirigido, na adição de cada aresta, (u, v) é igualmente adicionada uma segunda no sentido inverso (v, u).

Antes de se processar o grafo na procura de uma árvore de expansão mínima, é verificada a sua conectividade. Um grafo é conexo quando existe sempre pelo menos um caminho entre quaisquer dois vértices distintos, salvo a excepção de o grafo apenas conter um vértice. Para resolver o problema inicial, recorreu-se a uma variante do algoritmo de Kruskal.

2.1 Algoritmo de Kruskal

O algoritmo de Kruskal é um algoritmo ganancioso que consegue descobrir um subconjunto acíclico que conecta todos os vértices por forma a ter um peso total mínimo. A este subconjunto também se pode designar por árvore de expansão mínima.

Inicializam-se todos os vértices como grupos separados e declara-se o número de arestas aceites como nulo. É feita uma cópia de todas as arestas existentes para um vector.

No ciclo principal, as arestas são ordenadas por ordem não decrescente de distância. As arestas mais pequenas vão sendo progressivamente seleccionadas. Caso os vértices u e v que compõem a aresta (u, v) façam parte do mesmo grupo, a aresta não pode ser adicionada, uma vez que a árvore formaria um ciclo.

Como pode haver mais do que uma central, o ciclo anterior é executado enquanto o número de arestas aceites for menor do que o número de vértices, menos o número de centrais. A árvore de expansão mínima resultante destas operações é armazenada num contentor com apontadores para vértices numa nova instância da classe `Graph`. Sendo o número de arestas $|E|$, o número de vértices $|V|$ e o número de centrais $|C|$, a complexidade temporal deste processo é de $O(|E| \log |V - C|)$.

É feita a detecção dos vértices que são intersecções desprezáveis por não fazerem ligação a mais do que uma casa, reduzindo ainda mais a distância total necessária para ligar todas as casas.

Caso apenas exista uma central, a área dada como *input* é considerada como último filtro a aplicar sobre as casas seleccionadas. Quando é feita a instalação de mais do que uma central, é feita uma pesquisa do vértice-casa mais centralizado nas sub-árvores de expansão mínima.

3 Lista de casos de utilização

A aplicação permite a leitura de vértices e arestas contidos em dois ficheiros e a construção de um grafo não dirigido que represente um mapa de uma aldeia, com o qual se pode calcular uma árvore de expansão mínima que contemple todas as casas.

- Atribuição de uma área máxima em m^2, limitando o número de vértices que são abrangidos na árvore de expansão mínima final.
- Adição de novas centrais e cálculo do seu posicionamento ideal.
- Sobre os grafos de entrada e saída, é possível consultar graficamente informação acerca do número de vértices e arestas e a distância total em metros, soma de todas as arestas. As intersecções entre ruas são mostradas como vértices amarelos, as casas como vértices azuis e as centrais a vermelho.
- Modo de visualização dos vértices com respectivos *id* e posição dada pelos valores x e y em pixeis e o seu tipo `INTERSECTION`, `HOUSE`, ou `CENTRAL`, 0, 1 e 2 respectivamente.
- Modo de visualização das arestas com respectivos *id* de origem e de destino e distância em metros como peso da aresta.
- Multi-plataforma, testado em sistemas Linux e Windows.

4 Relato das principais dificuldades encontradas no desenvolvimento do trabalho

Sentiram-se dificuldades na compreensão do enunciado e na definição de fórmulas equivalentes aos dados de entrada pretendidos.

Embora numa fase inicial tenhamos usado o algoritmo de Prim para determinar uma árvore de expansão mínima, em consequência da necessidade de calcular para várias centrais, adaptámos mais rapidamente o algoritmo de Kruskal para este efeito.

Apesar de termos tentado determinar as melhores posições para as várias centrais, quando estas não são fornecidas nos dados de entrada, não conseguimos defini-las por forma a abranger o maior número de casas possíveis.

Ao contrário de uma utilização em Windows relativamente sem problemas, o código fornecido para Linux do visualizador de grafos não mantinha o processo-pai vivo, após o fecho da janela resultante da chamada `fork()`, pelo que esta parte teve que ser rescrita, instalando-se um `SIGCHLD` *handler* que faz uma chamada da função de sistema `waitpid()`.

5 Indicação do esforço dedicado por cada elemento do grupo

Planeou-se o esqueleto do projecto e implementou-se a leitura de ficheiros em conjunto.

5.1 José Peixoto

- Responsável pela interface gráfica e da linha de comandos.
- Incorporação do código desenvolvido nas aulas práticas.
- Verificação da conectividade dos grafos.
- Limpeza das intersecções desprezáveis.
- Autor do relatório.

5.2 Pedro Moura

- Adaptação do algoritmo de Kruskal.
- Selecção de centrais em quantidade indicada pelo utilizador.
- Limpeza das intersecções desprezáveis.

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

A Map

Rio Vez
EN 202
Casa Real
Rua Doutor Mário Tavarela Lobo
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
141 m
142 m
145 m
92 m
122 m
111 m
80 m
196 m
203 m
251 m
220 m
250 m
104 m
345 m
221 m
124 m
481 m
© OpenStreetMap contributors ♥ Make a Donation
50 m
200 ft
Search
Where am I?
Go

Referências

[1] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, *Introduction to algorithms Third Edition*, Cambridge, MA [etc.] : The MIT Press, cop. 2009.