Análise e Síntese de Algoritmos

Relatório do 2º Projecto

Introdução

A empresa de transporte de mercadorias, Coelho e Caracol Lda., pretende descobrir qual a melhor localidade para organizar um encontro de empresa, que deverá juntar os vários empregados das diversas filiais.

Uma rota é uma sequência de localidades e entre cada par de localidades pode ser transportado um tipo de mercadoria. O transporte gera receita e a deslocação gera um custo.

Para minimizar os custos das rotas até ao ponto de encontro, os empregados podem fazer entregas ao longo do percurso. O Sr. Caracol simplifica esta complexidade associando um valor de perda a cada par de localidades, que consiste em subtrair a receita ao custo.

Com este projecto pretendemos, dada a existência de várias filiais, descobrir a localidade em que a perda total da empresa seja minimizada.

Descrição da Solução

De modo a solucionar o problema, modelámos o mesmo sob a forma de um grafo dirigido e pesado. As localidades correspondem aos vértices do grafo. Um arco entre dois vértices representa a possibildade de fazer um transporte entre essas localidades. O peso de um arco é o respectivo valor de perda.

A implementação da solução foi concebida utilizando a linguagem de programação C, com a qual implementámos o grafo recorrendo a listas de adjacências. Esta implementação consiste em manter, para cada vértice do grafo, uma lista de todos os outros vértices com os quais este partilha uma aresta e o peso da aresta entre esse vértice e a respectiva adjacência. A direção é representada através da presença de um nó na lista de um vértice (a indireção seria representada

através da presença de um nó na lista correspondente à adjacência de mesmo vértice).

Este problema reduz-se a encontrar o caminho mais curto de fontes múltiplas a uma determinada localidade, onde seria o ponto de encontro óptimo. Como tal, decidimos aplicar às filiais em questão o algoritmo de Bellman-Ford, que consiste em encontrar o caminho mais curto com fonte única para qualquer outro vértice.

Para identificar caminhos mais curtos com fonte única (*SSSP*s), o Algoritmo de Bellman-Ford permite a existência de pesos negativos, ao contrário do Algoritmo de Dijkstra, e identifica a existência de ciclos negativos. É um algoritmo baseado no princípio de relaxação de arcos e apenas requer manutenção da estimativa associada a cada vértice. No entanto, é mais lento, comparado com o Algoritmo de Dijkstra, mas vantajoso e versátil dado o problema em questão (o valor de perda poder ser negativo, i.e., pesos de arcos negativos).

A ideia consiste em aplicar o algoritmo a cada filial, acumular os custos dos caminhos das fontes (filiais) a todos os vértices e minimizar esse valor, descobrindo assim o ponto de encontro. Descoberto o ponto de encontro, ainda são requeridos os custos individuais por filial, os quais podem ser recuperados aplicando uma iteração de Bellman-Ford ao anti-grafo a partir do ponto de encontro.

Análise Teórica

Pseudo-código do algoritmo de Bellman-Ford implementado[1]:

```
Bellman-Ford(G,s)
 1. for v ∈ V[G]   /* Initialize single source */
 2.     do d[v] = ∞
 3. d[s] = 0
 4. for i = 1 to |V[G]| - 1
 5.   changes = FALSE
 6.   do for each (u,v) ∈ E[G]
 7.      do if d[v] > d[u] + w(u,v)   /* Relax edge (u,v), if needed */
 8.         d[v] = d[u] + w(u,v)
 9.         changes = TRUE
10.   if changes is FALSE do break  /* If no changes were made, stop */
11. return d
```

1 A verificação de ciclos negativos não é feita uma vez que sabemos *a priori* que estes não existem (descriminado no enunciado do projecto).

A complexidade do algoritmo Bellman-Ford, no pior caso, é $O(VE)$ pois relaxa todas as arestas em E pelo menos $|V|$ - 1 vezes. No melhor caso, quando não foram efectuadas operações de relaxação sobre nenhuma aresta, o algoritmo toma a complexidade $O(E)$.

Seja S o conjunto das filiais. O algoritmo de Bellman-Ford é aplicado por filial de modo a obter e acumular os custos. Por cada iteração, executamos ainda um ciclo para fazer essa acumulação de modo a obter as somas dos custos num outro vetor x de dimensão V. Utilizando este vetor, é possível determinar o ponto de encontro, obtendo o menor custo ou menor distância a um determinado vértice.

Reconhecido agora o ponto de encontro, basta aplicar o Algoritmo de Bellman-Ford ao anti-grafo com fonte no ponto de encontro, recuperando as distâncias ou custos às filiais para obter os custos por filial (É trivial perceber que, dado um caminho $p_k = <v_0, \ldots, v_k>$ num grafo G, existe o respectivo "anti-caminho" $p_k^{-1} = <v_k, \ldots, v_0>$ no anti-grafo G^T sendo o peso do arco (v_i,v_{i+1}) em G igual ao do "anti-arco" (v_{i+1},v_i), em G^T, para $i \in [0, k - 1]$. Como tal, o caminho do ponto de encontro às filiais terá o mesmo custo que das filiais ao ponto de encontro, tornando excusada a aplicação do algoritmo a cada uma das filiais uma segunda vez de modo a obter os custos que são necessários, requerendo consequentemente menos espaço para armazenar informação).

Concluindo, aplicando Bellman-Ford $|S| + 1$ vezes (às filiais e ao ponto de encontro, no anti-grafo), a determinação dos valores de d e consequente acumulação em x $|S|$ vezes (1 por iteração de Bellman-Ford, i.e., por filial), determinação do ponto de encontro (linear em $|V|$), obtemos uma complexidade de $O((S + 1)VE + SV + V)$ total, no pior caso (ignorando os tempos de alocação e inicialização de recursos como o grafo e o vetor x e sua consequente libertação).

Avaliação Experimental

Inicialmente, sujeitámos a solução a quatro testes disponibilizados pelo corpo docente, tendo aparentemente passado aos testes sem problemas, uma vez que o output do programa não diferia do esperado. Repetimos o procedimento quando foram disponibilizados os primeiros seis testes utilizados pelo sistema de availação *Mooshak*, consequentemente obtendo resultados igualmente positivos.

De seguida, fizemos a submissão no sistema de avaliação, o qual sujeitaria a solução aos testes que nos foram fornecidos (primeiros seis) acrescido de testes mais rigorosos (dez seguintes), obtendo resultados satisfatórios, tendo resultado positivo num total de dezasseis testes.

Referências Bibliográficas

- Introduction to Algorithms, Third Edition. Chapter 24: Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein. September 2009. ISBN-10: 0-262-53305-7; ISBN-13: 978-0-262-53305-8. Last consulted April 21st 2016;
- Análise e Síntese de Algoritmos, 2º Semestre 2015/2016. Slides das aulas teóricas: Caminhos mais curtos com fonte única [CLRS, Cap. 24] *in* https://fenix.tecnico.ulisboa.pt/downloadFile/563568428735969/ch24.pdf. Last consulted April 23rd 2016;
- Bellman-Ford Algorithm: Wikipedia, the free encyclopedia *in* https://en.wikipedia.org/wiki/Bellman%E2%80%93Ford_algorithm. Last consulted April 23rd 2016, when article was modified on March 22nd 2016 at 20:18 UTC.