Relatório do 2º projecto de ASA

Rodrigo André Moreira Bernardo
ist178942

Instituto Superior Técnico

24 de Abril de 2015

1 O Problema

1.1 Introdução

A empresa Coelho e Caracol, Lda. faz transporte de mercadorias. O Sr. Coelho trata do transporte, enquanto o Sr. Caracol fica no escritório a fazer o planeamento das rotas, sendo que cada uma consiste numa sequência de localidades. Cada conexão entre localidades tem a si associada um valor de perda, que resulta de subtrair a receita ao custo.

1.2 Objectivo

Dado um input que identifique a sede da empresa, o número de localidades e as conexões entre si, assim como o valor de perda de cada uma, determinar a perda mínima desde a sede até cada localidade. O output deverá indicar para cada localidade o respectivo valor de perda. No entanto, caso uma localidade seja impossível de alcançar, então o seu valor de perda deve ser "U" e, caso seja impossível definir essa perda, então dizemos que o seu valor de perda é "I".

2 A Solução

O programa foi implementado em linguagem C. A solução passa por executar o algoritmo de Bellman-Ford para caminhos mais curtos sobre um grafo, G, cujos vértices representam as localidades e cujas arestas representam as

conexões entre as localidades. No último ciclo do algoritmo, caso se verifique que ainda seria possível relaxar uma determinada aresta (u, v), ou seja, se existir ciclos negativos, então é feita uma procura em largura primeiro (BFS) com início em v, marcando a estimativa de caminho mais curto de cada vértice visitado a $-\infty$, o que significa que é um vértice "I".

2.1 A Representação

Cada localidade é representada por um inteiro. A representação do grafo é em listas de adjacências. Mais pormenorizadamente, um grafo é uma estrutura composta por um vector de ponteiros para vértices e por um inteiro que indica o seu número de vértices (linhas 57-60). Cada vértice é uma estrutura com vários elementos, entre eles uma chave que o representa e um ponteiro para uma aresta adjacente (linhas 9-20). Cada aresta depois tem um ponteiro para a aresta adjacente seguinte (linhas 26-34). Propriedades características de cada algoritmo (como estimativa de caminho mais curto no algoritmo de Bellman-Ford ou cor na BFS) estão nas estruturas dos vértices e das arestas (em vez de existirem vectores para cada propriedade) para ser mais fácil aceder a essas propriedades nas várias partes do programa.

Para representar $-\infty$ e $+\infty$, foram utilizados INT_MIN e INT_MAX, respectivamente. Para acomodar esta escolha, foi criada uma função de soma (linhas 254-258), de forma a evitar *overflows*.

2.2 O Algoritmo

O algoritmo é uma versão modificada do algoritmo de Bellman-Ford. No entanto, foram feitas duas optimizações. A primeira passa por terminar o algoritmo caso tenha havido uma iteração onde não se verificasse relaxamento de arestas. A segunda passa por, a cada iteração, relaxar apenas arestas cujo vértice da qual partem tenho visto a sua estimativa de caminho mais curto alterada desde a última vez que essas arestas foram relaxadas.

Em relação à detecção dos vértices "I", é também verificado se a estimativa de caminho mais curto de cada vértice encontrado não é já $-\infty$. Se for, termina-se a BFS (ou nem sequer se inicia), de forma a diminuir o número de pesquisas redundantes.

3 Análise Teórica

Sejam V e E o número de vértices e arestas do grafo, respectivamente. O programa começa por ler as duas primeiras linhas do input, obtendo N, o número de localidades, C, o número de custos, e H, um inteiro que representa a sede da empresa. Depois inicializa o grafo com V=N vértices, que demora O(V). Ler as linhas restantes para adicionar as arestas é O(E), tendo em conta que adicionar uma nova aresta ao grafo é O(1), E=C, e considerando que as operações de ler linhas e alocar memória são O(1). Até agora temos uma complexidade para o tempo de O(V + E).

O algoritmo (modificado) de Bellman-Ford começa por inicializar todos os vértices, que é O(V). A seguir tem um ciclo com, no máximo, V-1 iterações, onde se aplica a operação de relaxar as arestas. No máximo, todos as arestas são relaxadas. Como relaxar um arco é O(1), temos que o ciclo é O(VE). Depois, se tiverem ocorrido relaxamentos na iteração V-1, o algoritmo verifica se existem ciclos negativos. O algoritmo de Bellman-Ford garante que após V-1 iterações de relaxamentos todos os vértices têm as suas estimativas de caminho mais curto correctas, a não ser que existam ciclos negativos. Nesta versão, quando se detectam ciclos negativos é feita uma BFS para atingir todos os vértices que o ciclo "atinge" e alterar a sua estimativa de caminho mais curto para $-\infty$. É feita apenas uma BFS para cada ciclo negativo que existir. Concluí-se que o conjunto de todas BFS tem complexidade temporal O(V + E).

Imprimir o output é simples de analisar. Tem-se um ciclo para percorrer todos os vértices. Considerando as operações de impressão como sendo O(1), imprimir todo o output é O(V).

No final tem-se O(V + E) + O(VE) + O(V) = O(VE) para o tempo.

Em relação à complexidade espacial, excepto algumas alocações independentes do tamanho do input (que são, portanto, O(1)), apenas se fazem alocações a estruturas de dados com tamanho linear em V e em E, pelo que a complexidade espacial é O(V + E).

4 Análise Experimental

Com recurso ao gerador fornecido, o *p2-gen*, foram criados 500 grafos, com um número de vértices entre 10 e 49910, sendo que, para cada grafo, se tem

E=2V. Foram obtidos os tempos totais de execução com a ferramenta *time* da *BASH* e a memória ocupada com a ferramenta *valgrind*.

Verifica-se que a memória ocupada é linear sobre V+E. Já em relação ao tempo de execução, o gráfico já não se assemelha tanto a uma linha, como se poderia esperar. No entanto, ao contrário da memória ocupada, o tempo de execução depende de factores externos, como a sobrecarga do computador.