Universidade Federal de Minas Gerais
Departamento de Ciência da Computação
Algoritmos e Estruturas de Dados III
2º Semestre de 2013

Trabalho Prático 4 - A mochila anda a cavalo

1 Introdução

Zambis, após muito trabalho e tempo gasto em responder todas as cartas, telegramas e e-mails de sua festa, resolveu descansar em sua fazenda. Nela ele gosta muito de colher pêras e guardá-las em sua mochila para comer depois sob a luz do luar. Cada pêra tem um peso w_i e uma qualidade q_i. Obviamente, Zambis gostaria de maximizar a qualidade das pêras que foram colhidas. Porém, sua mochila não tem capacidade infinita. De fato, ela não suporta mais do que W quilogramas de pêra.

Ciente do seu excelente trabalho na ordenação das mensagens da festa da piscina, Zambis não larga do seu pé e resolve pedir a sua ajuda para resolver esse problema.

Mas a colheita de Zambis é um pouco diferente.

A colheita das pêras é feita a cavalo. Existem T árvores diferentes e cada árvore t_i produz pêras de peso w_i e qualidade q_i em quantidade ilimitada. Zambis é livre para colher quantas quiser, desde que elas caibam na mochila. Como o terreno é acidentado, não é possível ir de qualquer árvore para qualquer árvore. De fato, para cada árvore t_i há um conjunto de árvores $N(t_i)$ que podem ser atingidas a partir dela. Tão pouco é possível colher pêras para todo sempre. Cada passagem de uma árvore i para uma árvore j requer que se percorra uma distância d_{ij} metros e o cavalo disponível não é capaz de andar por mais que D metros no total sem ficar exausto.

Será que você consegue encontrar o caminho que maximiza a qualidade total das pêras que foram colhidas?

2 Solução

Para este trabalho, você deverá apresentar uma solução que utilize tanto o paradigma de programação dinâmica quanto paradigma da programação paralela.

Seu algoritmo deve disparar *Nthreads* sendo que cada uma delas é responsavel por avaliar um subproblema gerado pela programação dinâmica. Lembre-se de que não pode haver duas *threads* vericando o mesmo subproblema, caso contrário sua paralelização não será justicada. Outro lembrete é de que cada *thread* retornará uma resposta. Isso, não significa que esta é a melhor resposta (caminho com a maior quantidade de pêras de boa qualidade que caibam na mochila). Você deve esperar que as outras *threads* retornem seus resultados. Assim será possível saber qual a melhor resposta.

3 Entrada

A entrada será um arquivo no seguinte formato: na primeira linha estará um número inteiro K que presenta quantas instâncias o arquivo possui. Na próxima linha começa a primeira instância.

Na primeira linha de cada instância contém o número de árvores T, a distância D, em metros, que o cavalo pode andar, o peso W da mochila em gramas e a quantidade C de caminhos que existem entre as árvores. A seguir, nas próximas T linhas estarão descritos a árvore t_i, o peso w_i e a qualidade q_i de suas pêras. A qualidade Q das pêras pode variar de 1(pior qualidade) até 10(melhor qualidade). Nas próximas C linhas está descrito a árvore t_i e t_j e a distância d_{ij} entre elas. Vale lembrar que, caso nessas C linhas não seja colocado a árvore t_x e t_y e sua distância d_{xy}, significa que não é possível alcançar t_y a partir de t_x e vice-versa Na próxima linha começa a próxima instância.

Os valores T,D,C e Q serão representados por um inteiro. Já a variável W será representada por um inteiro.

Veja o exemplo a seguir:

```
2
6 10 310 7
1 70 2
2 100 3
3 20 7
4 90 4
5 20 3
6 10 1
1 2 3
1 5 2
2 3 4
2 4 2
3 6 3
4 5 3
4 6 5
5 6 317 6
1 10 10
2 30 9
3 1 1
4 3 6
5 2 4
1 2 3
1 3 2
1 4 2
2 3 1
3 5 3
4 5 4
```

4 Saída

A saída deve mostrar o caminho que foi percorrido para conseguir maximizar a qualidade das pêras coletadas. Entre os caminhos de duas instâncias deve haver uma linha em branco. Para cada caminho, imprima a qualidade total da mochila seguida pelas tuplas: árvore visitada e a quantidade de pêras colhidas na árvore.

No primeiro exemplo abaixo, a visita começa na árvore 4 onde nenhuma pêra é colhida, segue para a árvore 6 onde é colhida apenas uma pêra e termina na árvore 3 onde 15 pêras são colhidas. A qualidade total da mochila é 106.

Note que pode haver mais de um caminho que maximize a qualidade das pêras colhidas. Não existe critério de desempate: escolha qualquer um dos caminhos que tenha a qualidade máxima.

Veja a saída para o exemplo anterior:
106 4 0 6 1 3 15
634 4 105 5 1

5 Entrega

- A data de entrega desse trabalho é **22/11/13**.
- A penalização por atraso obedece à seguinte fórmula $2^{d-1}/0.32\%$, onde d são os dias úteis de atraso.
- Submeta apenas um arquivo chamado <número matricula>_<nome>.zip. Não utilize espaços no nome do arquivo. Ao invés disso utilize o caractere '_'.
- Não inclua arquivos compilados ou gerados por IDEs. **Apenas** os arquivos abaixo devem estar presentes no arquivo zip.
  - Makefile
  - Arquivos fonte (*.c e *.h)
  - Documentacao.pdf
- Não inclua **nenhuma** pasta. Coloque todos os arquivos na raiz do zip.
- Siga rigorosamente o formato do arquivo de saida descrito na especificação. Tome cuidado com whitespaces e formatação dos dados de saída
- NÃO SERÁ NECESSÁRIO ENTREGAR DOCUMENTAÇÃO IMPRESSA!
- Será adotada **média harmônica** entre as notas da **documentação e da execução**, o que implica que a nota final será 0 se uma das partes não for apresentada.

6 Documentação

A documentação não deve exceder 10 páginas e deve conter pelo menos os seguintes itens:

- Uma **introdução** do problema em questão.
- **Modelagem e solução** proposta para o problema. O algoritmo deve ser explicado de forma clara, possivelmente através de pseudo-código e esquemas ilustrativos. **Lembre-se de mostrar que o problema pode ser resolvido utilizando programação dinâmica**, isto é, que ele possui subestrutura ótima e sobreposição de problemas.
- **Análise de complexidade** de tempo e espaço da solução implementada.
- **Experimentos e análise de rsultados** variando-se o tamanho da entrada e quaisquer outros parâmetros que afetem signicavelmente a execução. Explique claramente o que significa cada resultado alcançado
- Especificação da(s) **máquina(s) utilizada(s)** nos experimentos realizados.
- Uma breve **conclusão** do trabalho implementado.

7 Código

O código deve ser obrigatoriamente escrito na **linguagem C**. Ele deve compilar e executar corretamente nas máquinas Linux dos laboratórios de graduação.

- O utilitário **make** deve ser utilizado para auxiliar a compilação, um arquivo *Makefile* deve portanto ser incluído no código submetido. O utilitário deverá gerar um executável com o nome **tp4** que deverá **obrigatoriamente** ser capaz de receber o nome do arquivo de entrada e do arquivo de saída. O comando para executá-lo deverá seguir o exemplo abaixo:
  /.tp4 input.txt output.txt
- As estruturas de dados devem ser **alocadas dinamicamente** e o código deve ser **modularizado** (divisão em múltiplos arquivos fonte e uso de arquivos cabeçalho .h)
- **Variáveis globais** devem ser evitadas.
- Parte da correção poderá ser feita de forma automatizada, portanto siga rigorosamente **os padrões de saída especificados**, caso contrário sua nota pode ser prejudicada.
- **Legibilidade e boas práticas** de programação serão avaliadas