Algoritmo de Roteamento

Teleinformática e Redes 2 (TR-2)

Jacir Luiz Bordim
Departamento de Ciência da Computação (CIC)
Universidade de Brasília (UnB)

22 de março de 2016

– Algoritmos de Roteamento

Esta tarefa visa estudar e implementar os algoritmos de roteamento considerados nos capítulos 04 e 02 dos livros do Kurose et al. [1] e do Medhi et al. [2], respectivamente. Os algoritmos a serem implementados serão distribuídos aos respectivos grupos em sala de aula. Cada grupo será formado de no máximo 2 (dois) alunos. Para cada algoritimo implementado, deve-se observar os seguintes requisitos:

1. A linguagem de programação: C/C++ (Linux de preferência). As funções (algoritmos de roteamento, devem ser implementados pelos grupos, ou seja, funções prontas não serão aceitas);

2. O programa deve receber como entrada (argumento de linha):

   (a) **Topologia:** descrita em um arquivo texto, no formato de **lista de adjacência**, contendo na primeira coluna o nó, na segunda coluna um vizinho direto e o custo, na terceira coluna, o segundo vizinho e o custo, e assim por diante. Exemplo:

   $$1 - 2[4]; 3[1];$$

   $$2 - 1[4], 3[2];$$

   $$3 - 1[1]; 2[3]$$

   (b) **Fonte/destino:** Para os casos que se apliquem, o nó origem e destino, devem ser informados via argumento de linha;

3. O programa deve fornecer como saída:

   (a) **Caminho**: Quando fonte e destino são informados, o algoritmo deve apresentar a lista de nós e pesos a serem seguidos para alcançar o destino;

   (b) **Matriz**: Para casos onde fonte e destino não são informados, o algoritmo deve fornecer a lista completa (Matriz) contendo o "custo" para cada nó e seus respectivos destinos considerados;

4. Validação: o trabalho deve acompanhar de pelo menos 3 topologias de exemplo. Cada topologia deve conter um número crescente de nós e enlaces (e seus respectivos custos). A topologia mínima deve conter

pelo menos 6 nós, a segunda pelo menos 12, a terceira pelo menos 18. Cada nó deve conter entre [2,5] vizinhos (i.e, $G = (V, \emptyset)$ ou grafos completos (K_n) não são considerados exemplos válidos).

5. Documentação:

   (a) Cada grupo deve informar as restrições (limitações) do programa. Toda a função, ou código que não seja declarações devem ser comentadas. Da mesma forma, cada programa deve conter informações de como compilar e executar.

   (b) A documentação pode ser fornecida via LEIA.TXT (como compilar, executar os exemplos e detalhar o formato da lista de adjacência, e demais parâmetros de entrada, restrições etc). As demais informações devem ser inseridas no código (comentários sobre a implementação do algoritmo).

6. Exemplos Prontos: Encontrar um exemplo da Internet, livro ou alguma outra fonte e adaptar é válido. Agora, você deverá informar a fonte que utilizou como exemplo (seja o livro que contém a descrição do algoritmo e/ou código utilizado como base). Esta informação é **obrigatória**. Da mesma forma, o custo computacional do seu algoritmo deve ser informado;

7. Cópia: Copiar do seu colega, adaptar do seu colega não será permitido. Como não saberei quem copiou de quem, todos serão penalizados, sem exceção.

8. Prazo de entrega: 1 semana (via aprender). O grupo deverá apresentar em sala o código e exemplos funcionando.

Lista dos algoritmos a serem implementados e os respectivos grupos

1. Dijkstra
2. Belmann-Ford
3. k-shortest path ($k-1$ variações do melhor caminho. Vide algoritmo de Yen's)
4. k-shortest disjoint path ($k-1$ variações do melhor caminho)
5. Widest path
6. Floyd-Warshall
7. Kruskal Algorithm (árvore geradora mínima, do Inglês *Minimum Spanning Tree* (MST))
8. Prims Algorithm (MST)
9. Implementar um algoritmo de roteamento que trata enlaces com peso negativo. Devem mencionar quais algoritmos podem ser utilizado para esta tarefa
10. **Desafio**: Implementar um algoritmo de roteamento distribuído, onde os nós não possuem informações completa da topologia, exceto dos nós vizinhos. Os nós podem se comunicar via troca de mensagens.

Referências Bibliográficas

[1] James F. Kurose and Keith W. Ross. *Computer Networking: A Top-Down Approach (6th Edition)*. Pearson, 6th edition, 2012.

[2] Deepankar Medhi and Karthikeyan Ramasamy. *Network Routing: Algorithms, Protocols, and Architectures*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007.