Rudini Menezes Sampaio
Orientador: Horácio Hideki Yanasse
Aluno de Engenharia de Computação do Instituto Tecnológico de Aeronáutica (ITA)
Laboratório Associado de Computação e Matemática Aplicada (LAC)
Instituto Nacional de Pesquisas Espaciais (INPE)
Bolsa PIBIC/CNPq
Avenida dos Astronautas, 1758 – Caixa postal 515

ESTUDO E IMPLEMENTAÇÃO DE PROBLEMAS COMBINATORIAIS EM GRAFOS

Existem vários problemas urbanos de fluxo que podem ser modelados como problemas combinatoriais em grafos. Entrega de correspondência, limpeza de rua, coleta de lixo, serviço de ônibus para melhor atendimento à população, reparos de cabines telefônicas, remoção de neve e problemas de seleção de lugares para localização de serviços de emergência (prontos-socorros, bombeiros,...) ou de não emergência (terminais de transporte,...) são exemplos de problemas práticos que podem ser modelados desta forma.

A teoria de grafos, direcionada para solução de tais problemas, possui várias vantagens, entre elas a fácil descrição do problema para parâmetros da teoria e a facilidade com que se pode encontrar soluções aproximadas (heurísticas) para problemas matematicamente complexos, cujas soluções ótimas e exatas são difíceis de se obter para um tempo hábil.

O rápido avanço na tecnologia dos computadores proporcionou uma atenção especial para esta teoria, ocasionando o desenvolvimento de vários algoritmos que solucionem esses problemas. Tais algoritmos resolvem problemas de menor caminho, árvore de custo mínimo, carteiro chinês, caixeiro viajante, entre outros.

Sendo assim, este trabalho tem por objetivo o estudo dos diversos problemas combinatoriais em grafos, suas soluções teóricas e heurísticas, as implementações destas em uma linguagem de computação, a execução destes em um ambiente "user-friendly" desenvolvido especialmente para facilitar a visualização e a manipulação de grafos, bem como a execução dos algoritmos construídos.

Para isso desenvolveu-se um software gráfico que realizasse tais requisitos, com a escolha adequada para estrutura de dados e entrada de grafos por arquivos externos e pelo estudo aprofundado de técnicas de engenharia de software e linguagens de programação como *Pascal* e *Assembly*.

Durante o desenvolvimento do software, preocupou-se bastante com a visualização na tela e a facilidade de utilização do mesmo, com a construção dos algoritmos e principalmente com a execução de testes de verificação e desempenho. Tais testes tiveram papel fundamental para o projeto, e por isso, muito do tempo destinado a ele foi gasto com a realização de tais testes.

Para desenvolvimento dos algoritmos, estudou-se os problemas a serem resolvidos, suas soluções teóricas e heurísticas, e escolheu-se a melhor opção a ser implementada. Desenvolveu-se algoritmos para resolução dos seguintes problemas combinatoriais em grafos:

- **Orientação/Direção** em um grafo, capaz de informar se um determinado grafo é direcionado ou não, ou seja, se seus arcos possuem direção ou tanto faz ir como vir. De grande interesse teórico e bastante útil para outros algoritmos.

- **Conectividade**, capaz de informar com precisão se determinado grafo é fortemente conexo, simplesmente conexo ou não é conexo. Ou seja, se de um nó é possível acessar todos os outros nós do grafo. De grande interesse teórico e bastante útil para futuros algoritmos, como no de *identificação de pontes* descrito posteriormente.

- **Paridade**, capaz de informar a paridade de todos os nós de um grafo, ou seja, se do nó sai um número par de arcos (grau par) ou um número ímpar de arcos (grau ímpar). É de fundamental importância para o algoritmo do *Matching Problem* descrito posteriormente.

- **Menor Caminho de *Dijkstra***, capaz de obter os menores caminhos de um dado nó do grafo para todos os outros nós. Útil, por exemplo, para casos de melhores rotas a serem seguidas numa viagem.

- **Menor Caminho de *Floyd***, capaz de obter as menores distâncias e seus percursos entre todos os nós de um grafo. Útil, por exemplo, para a formação de tabelas rodoviárias que indicam as distóancias entre cidades de um estado ou região. Juntamente com o algoritmo de *Dijkstra*, é de fundamental importância para o algoritmo da *Árvore de Custo Mínimo*, *Carteiro Chinês* e *Caixeiro Viajante*, descritos a seguir.

- **Árvore de Custo Mínimo (*Minimum Spanning Tree*)**, capaz de obter em um grafo dado, uma árvore cuja soma dos nós é mínima. Útil, por exemplo, no auxílio à decisão de onde se implantar postos de emergência em uma comunidade. De fundamental importância para o algoritmo do *Caixeiro Viajante*.

- **Matching Problem**, capaz de obter o conjunto dos pares de nós de grau ímpar de um grafo cuja soma dos menores caminhos entre os nós de cada par é mínima. De grande interesse teórico e parte integrante da solução do problema do *Carteiro Chinês*.

- **Identificação de Pontes (*Bridges*)** em um grafo, capaz de informar se determinado arco é uma ponte, ou seja, com a sua eliminação o grafo torna-se desconexo. Útil, por exemplo, em uma guerra na decisão de que pontes se destruir para deixar o inimigo isolado, ou na escolha dos lugares de se realizar emboscadas. De primordial utilidade para o algoritmo de *Formação de Ciclo Euleriano*, descrito a seguir.

- **Formação de Ciclo *Euleriano***, capaz de obter o ciclo ou circuito *euleriano* de um determinado grafo que não contenha nós de grau ímpar. Ou seja, saindo de um nó percorre-se todos os arcos do grafo retornando no fim ao nó inicial. Consiste dos algoritmos de *Conectividade*, *Paridade* e de *Identificação de Pontes*. De grande interesse teórico e parte integrante do problema do *Carteiro Chinês*.

- **Carteiro Chinês (*Chinese Postman*)**, capaz de obter a melhor rota (custo mínimo) a ser seguida com o objetivo de percorrer todos os arcos de um grafo ao menos uma vez e retornar ao nó inicial. Útil, por exemplo, para os Correios e para o Serviço de Limpeza Pública. A solução desse problema consiste da utilização de dois algoritmos: *Matching Problem* e *Formação de Ciclo Euleriano*, ambos descritos anteriormente.

- **Caixeiro Viajante (*Traveling Salesman*)**, capaz de obter a melhor rota a ser seguida com o objetivo de se visitar todos os membros de um conjunto de nós específico do grafo ao menos uma vez, retornando eventualmente ao nó inicial. Útil, por exemplo, para uma empresa que faz entregas à domicílio ou à outras lojas. Implementou-se sua versão que inclui total conectividade entre todos os nós do grafo e desigualdade triangular nos arcos. A solução heurística desse problema consiste da utilização dos algoritmos do *Menor Caminho de Floyd*, *Árvore de Custo Mínimo* e *Carteiro Chinês*.

Bibliografia:

1. Aho, A.V.; Hopcroft, J.E.; Ulman, J.D., **The Design and Analysis of Computer Algoritms**, Addison-Wesley, 1974.
2. Boaventura Netto, P.O., **Grafos: Teoria, Modelos e Algoritmos**, Edgard Blücher, 1996.
3. Campello, R.E.; Maculan, N., **Algoritmos e Heurísticas**, EDUFF, 1994.
4. Duncan, R., **Advanced MSDOS Programming**, Microsoft Press, 1986.
5. Larson,R.C.; Odoni,A.R., **Urban Operations Research**, Prentice Hall,1981.
6. Mokarzel, F.C., **Apostila do Curso de Estrutura de Dados (CES-20)**, Computação, ITA, 1990.