FEMG - uma ferramenta prática, multiplataforma, interativa e acessível via web para manipulação de grafos

André Gustavo dos Santos, Matheus Júlio de Oliveira, Thiago Henrique Rodrigues

Departamento de Ciência da Computação – Faculdades Integradas de Caratinga (FIC)
Rua João Pinheiro, 168 – Centro – 35.300-037 – Caratinga – MG – Brasil
{andre, matheus, thiagoh}@spep.com.br

Abstract. *FEMG is an interactive web tool to solving all classical problems in graphs, such as traversing graphs, trees, network flows and coloring. Its main use will be to solve exercises or homework about graph theory, graph algorithms, and its applications. One semester is not long enough to ask for students to implement all such algorithms, but it is very important that they learn how to use graph to model a problem and solve it using an appropriate algorithm. When using FEMG, they can focus on the problem solving, helping they develop their graph modeling skills, instead of passing hours debugging code. They will not only model the problem using graphs, but they could also see the numerical results of their graph modeling, completely solving it.*

Resumo. *FEMG é uma ferramenta prática, interativa, acessível via web, funcional em qualquer sistema operacional e navegador, que resolve todos os problemas clássicos envolvendo grafos, tais como caminhamentos, árvores, fluxos e coloração. Diversos exercícios didáticos envolvendo teoria de grafos requerem a modelagem em grafos de algum problema real. Para problemas de tamanho maior, a solução à mão pode ser impraticável, e a implementação de todo o código demandaria tempo razoável do aluno, que passaria a concentrar esforços na implementação. Essa ferramenta permite encontrar a solução do problema no grafo modelado, mantendo o enfoque do aluno no assunto principal, que seria a utilização de grafos na resolução de problemas.*

1. Introdução

FEMG - *FE*rramenta para *M*anipulação de *G*rafos - é uma ferramenta *web* para execução de algoritmos em grafos, sendo assim acessível de qualquer computador ligado à internet, a partir de qualquer sistema operacional e navegador. Com uma interface de fácil utilização, permite a entrada dos dados do grafo de forma interativa ou através de arquivos, e execução de vários algoritmos, como caminho mínimo, árvore mínima, caixeiro viajante, matching, fluxos, entre outros.

FEMG serve como apoio ao ensino de grafos, na medida em que problemas práticos e reais podem ser modelados através de grafos e resolvidos através de sua utilização. Nossa experiência mostra que em um semestre de ensino de uma disciplina de grafos, os alunos não têm tempo suficiente para implementar todos os algoritmos clássicos, e por isso, em cada semestre apenas alguns desses algoritmos chegam a ser implementados. Para não restringir o aprendizado dos demais problemas e algoritmos, mostramos exercícios com problemas reais modelados e resolvidos através dos demais

conceitos e algoritmos em grafos. E os grafos obtidos nesse processo de modelagem podem ser resolvidos pelos algoritmos implementados pelo aluno ou utilizando o FEMG.

FEMG pode então ser usada por alunos que desejam verificar se o modelo feito resolve o problema prático modelado de forma satisfatória, ou conferir resultados de exercícios feitos manualmente, ou ainda testar algoritmos por eles implementados. Para os professores, constitui uma ferramenta auxiliar para rápida verificação da correção de dados enquanto prepara listas de exercícios, trabalhos ou provas, e ainda meio de conferir soluções dadas pelos alunos.

Através de extensa pesquisa na *web*, FEMG é uma das poucas ferramentas que engloba todos os diversos algoritmos clássicos sobre grafos, e ainda alguns outros relacionados. Possui ainda opção de utilização de heurísticas quando o tamanho do grafo torna impraticável a solução exata. E é a única acessível diretamente via web.

2. Outras ferramentas semelhantes

Existem outras ferramentas semelhantes a FEMG. Sua implementação se justifica pois várias das ferramentas existentes implementam apenas alguns algoritmos, a maioria necessita instalação em sistemas operacionais específicos, e as melhores são pagas, onerando custos dos alunos, ou da instituição que queira disponibilizá-la.

Através de busca na *web* em *sites* de busca, projetos de universidades, anais de congressos e simpósios da área, e indicação de professores que trabalham este conteúdo em sala da aula, selecionamos algumas ferramentas para análise comparativa, e também como fonte de inspiração para novas características de FEMG. As ferramentas e páginas *web* selecionadas foram:

- Netsolve: Desenvolvida por James Jarvis e Douglas Shier, acompanhava o livro de Evans e Minieka (1992)
- GRIN - Graph Interface: Desenvolvida por Vitaly Pechenkin, é uma das ferramentas atuais mais completas para manipulação de grafos. Suas versões gratuitas e com licença podem ser obtidas no site http://www.geocities.com/pechv_ru/
- DSPA - Dijkstra's Shortest Path Algorithm: Desenvolvida por Harvey J. Greenbeng, é uma ferramenta *web* para o problema de caminho mínimo. Site: http://carbon.cudenver.edu/~hgreenbe/sessions/dijkstra/DijkstraApplet.html

O Netsolve é um interpretador de comandos bastante completo para manipulação de grafos, oferecendo ao usuário a opção de carregar, editar e salvar um grafo, e vários algoritmos. Porém, deixa a desejar no que diz respeito à facilidade de uso por não ser uma ferramenta atual, desenvolvida para o sistema operacional DOS, e não possuir uma interface gráfica e intuitiva. Todas as tarefas são executadas via linha de comando, necessitando memorizar os comandos e os parâmetros, ou acessar, também via linha de comando, várias vezes, o menu de ajuda.

Já o GRIN possui uma interface gráfica eficiente na criação e visualização de grafos e oferece diversos algoritmos para manipulação. Apesar disso, possui o problema de ser uma ferramenta paga e sua versão gratuita possui muitas limitações. Além ainda de ser trabalhosa a criação de grafos extensos, com muitos vértices e arestas, já que não permite editar um arquivo de entrada.

DSPA é um Applet Java que resolve o problema de caminho mínimo entre vértices. Possui a vantagem de ter interface gráfica, permitindo inserir vértices, arestas, e modificar pesos visualmente, diretamente no desenho do grafo, e ainda ser acessada pela internet, mas sua capacidade de operação é muito restrita. já que só resolve o problema de caminho mínimo.

Como características principais para efeito de análise comparativa consideramos a facilidade de utilização, recursos de entrada e saída, algoritmos implementados, interface, limites, restrições, ajuda, instalação, entre outras. O quadro abaixo resume as principais características de cada ferramenta. Foi analisada apenas a versão livre e gratuita da ferramenta GRIN.

Quadro 1: quadro comparativo das ferramentas

Característica	Netsolve	DSPA	GRIN	FEMG
Salvar / Carregar grafo em arquivo	Sim	Não	Sim	Sim
Edição do arquivo de entrada	Sim	Não	Não	Sim
Edição interativa do grafo	Sim	Sim	Sim	Sim
Entrada gráfica (visual)	Não	Sim	Sim	Não
Exibição dos resultados graficamente	Não	Sim	Sim	Não
Execução passo-a-passo	Não	Sim	Não	Não
Tópicos de ajuda	Sim	Sim	Sim	Sim
Multiplataforma	Não	Sim	Não	Sim
Multiusuário	Não	Sim	Não	Sim
Sistema completo gratuito	Não	Sim	Não	Sim
Portabilidade	Não	Sim	Não	Sim
Utilização de arestas dirigidas	Sim	Sim	Não	Sim
Utilização de arestas não dirigidas	Não	Não	Sim	Sim
Algoritmo para Árvore Mínima	Sim	Não	Sim	Sim
Algoritmo para Coloração de Vértices	Não	Não	Sim	Sim
Algoritmo para Matching Mínimo/Máximo	Sim	Não	Não	Sim
Algoritmo para Fluxo de Custo Mínimo	Sim	Não	Sim	Sim
Algoritmo para Fluxo Máximo	Sim	Não	Sim	Sim
Algoritmo para Ciclo / Caminho Hamiltoniano	Não	Não	Sim	Sim
Algoritmo para Circuito / Caminho Euleriano	Não	Não	Sim	Sim
Algoritmo para Caminho Mínimo	Sim	Sim	Sim	Sim
Algoritmo para Caixeiro Viajante	Sim	Não	Sim	Sim
Heurística para Caixeiro Viajante	Sim	Não	Não	Sim
Algoritmo para Assinalamento	Sim	Não	Não	Não

3. A interface de FEMG

A interface de FEMG foi criada com o objetivo de tornar funcional e fácil sua utilização, sem exigir grande poder de comunicação pela *web* por parte dos usuários. Utilizando a linguagem de marcação HTML, criou-se uma interface principal leve e enxuta, que não exige muitos recursos gráficos do usuário e que minimiza o tráfego na rede. Através de um menu único podemos inserir os dados do grafo e solucionar diversos problemas, visualizando os resultados tão logo tenham sido encontrados. Assim, toda a manipulação de grafos é feita através de um menu principal, por onde são realizadas as operações:

- Requisição de um novo grafo (o FEMG trabalha com um grafo de cada vez);
- Carregar um grafo através de um arquivo
- Salvar o grafo em um arquivo
- Inserir / retirar vértices
- Inserir / retirar arestas
- Resolver problemas manipulando um grafo

A figura 1 mostra a interface principal de FEMG, como é vista do Netscape 7.1 no sistema Suse 8.0 KDE 3.2. A mesma interface será vista de outros navegadores e sistemas operacionais. Foi implementada em PHP, HTML e JavaScript [Ramalho, 1999, Thomson e Welling, 2001].

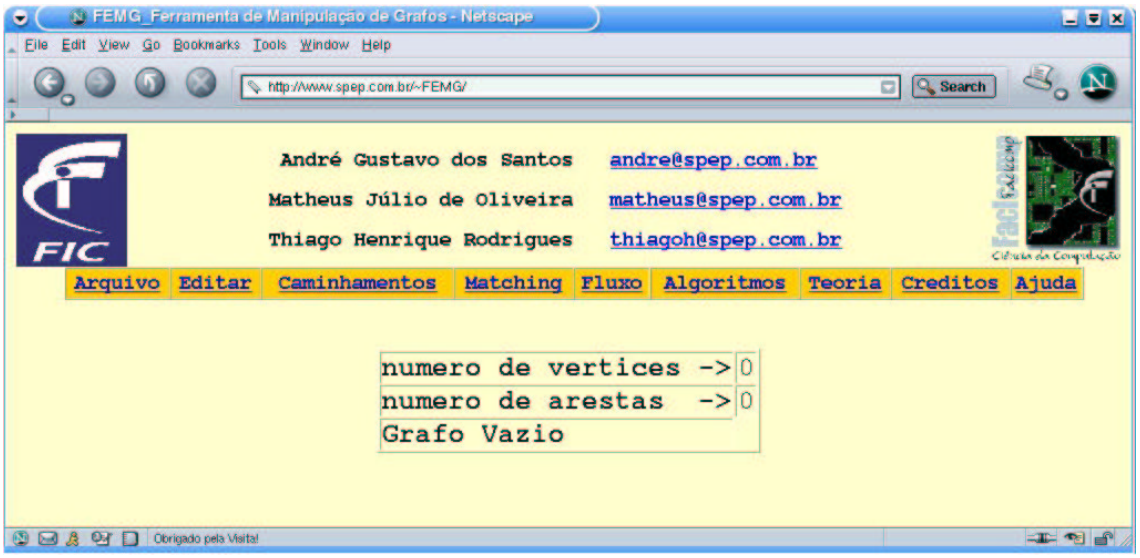


Figura 1: interface principal do FEMG

3.1. Entrada de dados

A entrada pode ser feita de duas formas: de forma interativa ou através de arquivos. A entrada via arquivo aumenta a flexibilidade do FEMG, permitindo que grafos maiores possam ser editados manualmente ou gerados por outros programas, e então lidos pelo FEMG. Desta forma, dados de problemas reais podem ser manipulados pelo FEMG, bastando-se converter os dados para o formato próprio, que consiste basicamente em se informar o número de vértices e arestas, os nomes dos vértices, e a lista de arestas. A opção Ver Estrutura do submenu Arquivo mostra o formato do arquivo texto usado pelo FEMG, como ilustrado no quadro a seguir.

Quadro 2: formato do arquivo de dados

`V`
`A`
`n1`
`n2`
`...`
`nV`
`vi1 vf1 p1`
`vi2 vf2 p2`
`...`
`viA vfA pA`
	• `V` = número de vértices do grafo
	• `A` = número de arestas do grafo
	• `ni` = nome do vértice `i`, para `i` = 1, 2, ..., `V`
	• `vij`, `vfj` = vértices inicial e final da aresta `j`, para `j` = 1, 2, ..., `A`. Esses valores devem estar entre 1 e `V`
	• `pj` = peso da aresta `j`, para `j` = 1, 2, ..., `A`

Na figura 2 mostramos um arquivo contendo os dados de um grafo para a distância entre as capitais do sudeste. Note que, como as arestas não são dirigidas, cada uma deve ser descrita duas vezes, uma em cada sentido. Note ainda que cada aresta na lista de arestas é representada por uma seqüência de três valores: índice do vértice de origem, índice do vértice de destino e o peso. Deve-se observar que o índice de um vértice corresponde exatamente à ordem informada anteriormente na lista dos nomes dos vértices, iniciando-se no índice 1.


```
4
12
BH
Rio
Sao Paulo
Vitoria
1 2 444
2 1 444
1 3 586
3 1 586
1 4 526
4 1 526
2 3 429
3 2 429
2 4 525
4 2 525
3 4 958
4 3 958
```

Figura 2. Entrada de dados via arquivo. (a) Mapa ilustrativo mostrando localização das capitais da região sudeste (b) arquivo com dados do grafo construído com as distância das capitais.

Os arquivos podem ser carregados para ser manipulado por FEMG através da opção Carregar do submenu Arquivo. Pode-se informar o arquivo desejado digitando seu nome com o caminho de acesso (navegação de diretórios) na caixa de texto apropriada, ou através do botão Procurar, que abre um *browser* para se navegar pela estrutura de diretórios e selecionar o arquivo desejado.

Já a entrada de dados de forma interativa oferece simplicidade para usuários que desejam criar grafos relativamente pequenos - como pode ser o caso de uso da ferramenta no meio acadêmico. Isso é feito através do submenu Editar do menu principal, onde se tem as opções de inserir ou retirar vértices e arestas (figura 3a). Os vértices devem ser inseridos antes das arestas e podem ser nomeados ou enumerados de acordo com a preferência do usuário (figura 3b). Já as arestas são definidas por caixas de texto que devem ser preenchidas com o peso da aresta indicada. Deixando uma caixa

de texto em branco, o usuário indica que o grafo não possui aresta com a origem e destino correspondentes. As caixas estão dispostas como uma matriz de adjacência, onde as linhas representam os vértices de origem e as colunas os vértices de destino (figura 3c). Existe ainda a oportunidade de se inserir arestas dirigidas ou não-dirigidas.

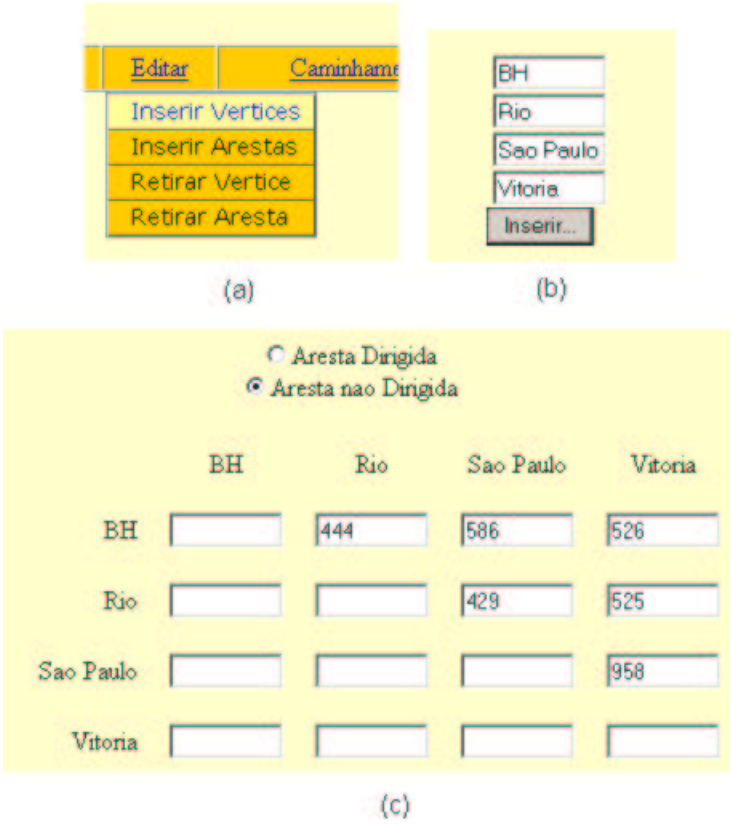
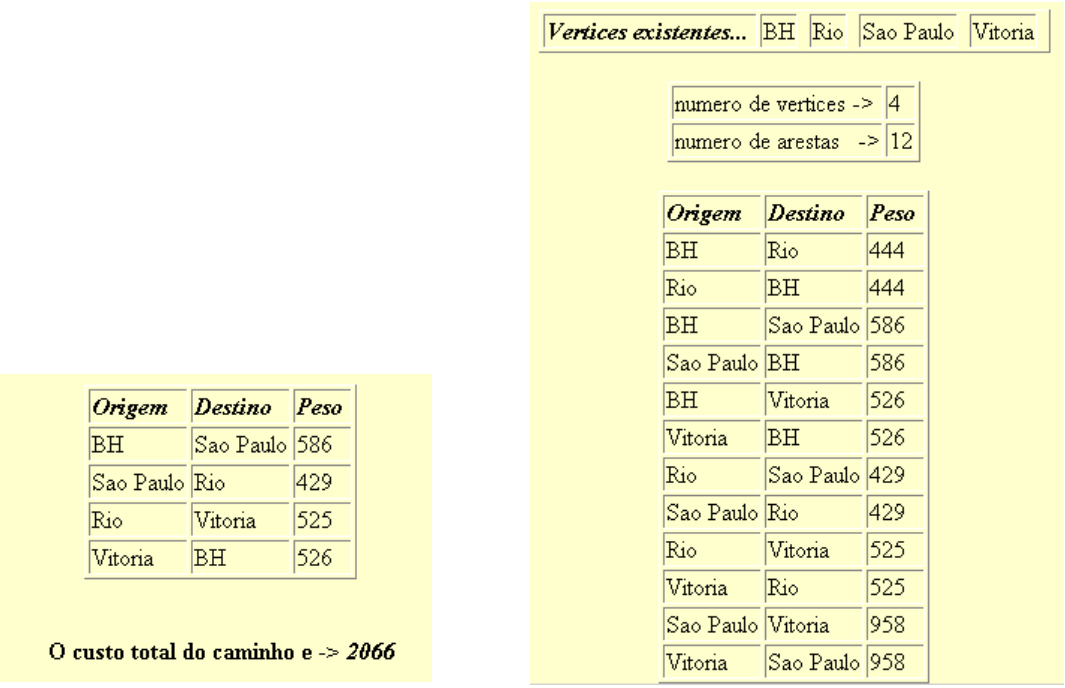


Figura 3. Entrada de dados interativa. (a) No menu escolhemos *Inserir vértices* (b) nomeamos os vértices (c) Escolhemos *Inserir arestas* e cadastramos o peso de cada uma

3.2. Saída de dados

Um grafo editado ou modificado em FEMG pode ser salvo de forma que possa ser utilizado posteriormente. A gravação é feita em formato texto obedecendo as especificações anteriores. Para isso, basta selecionar Salvar no submenu Arquivo. Será aberto um *browser* do sistema operacional para que o usuário possa navegar e selecionar o diretório onde deseja salvar o arquivo.

FEMG mantém o grafo em uso sempre visível na parte inferior da tela, priorizando a visualização dos resultados de sua manipulação. Ao aplicar qualquer dos algoritmos de FEMG no grafo, a visualização é automaticamente atualizada e os resultados da aplicação são instantaneamente visualizados para evitar problemas de tráfego na *web*. Por exemplo, para o grafo das capitais do sudeste mostrado anteriormente, ao se selecionar a opção de Caixeiro Viajante, a resposta será mostrada como na figura 4a. Logo abaixo, virá os dados originais do grafo corrente, como na figura 4b.



Origem	*Destino*	*Peso*
BH	Sao Paulo	586
Sao Paulo	Rio	429
Rio	Vitoria	525
Vitoria	BH	526

O custo total do caminho e -> *2066*

Vertices existentes...	BH	Rio	Sao Paulo	Vitoria

numero de vertices ->	4
numero de arestas ->	12

Origem	*Destino*	*Peso*
BH	Rio	444
Rio	BH	444
BH	Sao Paulo	586
Sao Paulo	BH	586
BH	Vitoria	526
Vitoria	BH	526
Rio	Sao Paulo	429
Sao Paulo	Rio	429
Rio	Vitoria	525
Vitoria	Rio	525
Sao Paulo	Vitoria	958
Vitoria	Sao Paulo	958

(a) (b)

Figura 4: (a) resultado do problema do caixeiro viajante (b) grafo corrente

FEMG também permite gravar o grafo resultante da aplicação de um algoritmo em arquivo texto com o mesmo formato definido na seção anterior.

4. Algoritmos implementados

FEMG contém os mais variados algoritmos para problemas em grafos. Listamos a seguir os problemas já suportados pelo FEMG e os algoritmos usados para resolvê-los. Todos os algoritmos implementados são algoritmos clássicos na solução desses problemas, e podem ser encontrados em praticamente todos os livros que tratam do assunto, como Cormen et al. (2001), Evans e Minieka (1992), Tucker (1994), Markenzon e Szwarcfiter (1994).

4.1. Caminhamento em grafos

Esses problemas envolvem caminhar no grafo através de caminhos de peso mínimo, caminhos que envolvem todas as arestas ou todos os vértices.

O caminho hamiltoniano consiste em se encontrar um caminho qualquer a partir de um determinado vértice que percorra todos os demais exatamente uma vez. Caso seja necessário terminar o caminho no mesmo vértice de início, desejamos um ciclo hamiltoniano. Pode-se pedir ao FEMG para encontrar um ou todos os caminhos (ciclos) hamiltonianos existentes. No problema do caixeiro viajante, o objetivo é encontrar o ciclo hamiltoniano de peso total mínimo. Todos esses problemas são resolvidos pelo FEMG através de técnicas de *backtracking*, ou seja tentativa e erro, já que não existem algoritmos polinomiais para tais problemas. Na solução do caixeiro viajante para grafos grandes, é oferecida ainda a opção de heurísticas polinomiais, que retornam uma solução não garantidamente ótima, porém em tempo satisfatório.

Em grafos eulerianos é possível encontrar um ciclo que percorre exatamente uma vez cada aresta do grafo. FEMG também encontra um caminho euleriano, que passa por todas as arestas exatamente uma vez, e que, diferentemente do ciclo não requer iniciar e terminar no mesmo vértice. Inicialmente as condições de existência de ciclos e caminhos são verificadas, uma tarefa fácil que consiste em verificar se o grafo é conectado e se todos os vértices possuem grau par (no caso de caminho euleriano é permitido que dois vértices tenham grau ímpar). Atendidas as condições, um algoritmo de geração de circuitos e posterior junção resolve o problema em tempo polinomial.

Outras vezes não há interesse em se percorrer todos vértices com custo mínimo, mas apenas os necessários para iniciar e terminar em certos vértices. O algoritmo de Dijkstra realiza essa tarefa polinomialmente, encontrando o menor caminho de um certo vértice a cada um dos demais. FEMG oferece a opção de encontrar o menor caminho entre dois vértices, entre um e todos os demais, e de mostrar não só o peso do menor caminho mas uma lista de vértices que indica o caminho.

4.2. Fluxo em redes

O problema de fluxo máximo ocorre em um grafo dirigido, onde o peso de cada aresta representa a sua capacidade de fluxo. O problema de fluxo máximo consiste em maximizar o fluxo entre dois vértices do grafo, nesse caso chamado rede, respeitando-se a capacidade de cada aresta da rede. O algoritmo implementado em FEMG é chamado *augmenting flow algorithm*, que consiste basicamente em se encontrar em cada iteração uma seqüência de arestas entre os vértices inicial e final cujo fluxo em cada aresta ainda não esteja no limite de capacidade. O fluxo é então aumentado nessa seqüência de arestas até que alguma aresta atinja o limite de capacidade. Essas seqüências são encontradas de forma semelhante ao algoritmo de caminho mais curto, e quando não houver mais nenhuma seqüência de arestas cujo fluxo possa ser aumentado, o algoritmo reporta a solução encontrada.

4.3. Outros problemas

FEMG também resolve problemas de árvore geradora mínima, coloração de vértices, e ainda problemas de emparelhamento (*matching*).

No problema da árvore geradora mínima estamos interessados em um árvore geradora do grafo, ou seja, um subgrafo conectado sem ciclos contendo todos os vértices do grafo, e cujo peso total das arestas escolhidas seja o mínimo possível. FEMG resolve este problema através do algoritmo de Prim, de complexidade polinomial. É um algoritmo "guloso", que constrói a árvore gradativamente, adicionando aresta por aresta, escolhendo sempre aquela de peso mínimo entre as que conectam um vértice ainda isolado na árvore já formada. Existe ainda um outro algoritmo polinomial bastante utilizado, proposto por Kruskal, porém o algoritmo de Prim funciona de forma semelhante ao algoritmo de Dijkstra já implementado para caminho mínimo, e por isso foi escolhido.

O problema de coloração consiste em, dado um grafo, determinar o número mínimo de cores a ser utilizadas para que cada vértice seja colorido com uma dessas cores, e que vértices adjacentes tenham cor diferente. Nesse caso os pesos das arestas não são utilizados. FEMG informa o número mínimo de cores e uma lista de cores dos

vértices, associando a cada vértice uma cor, numerada de 1 até o limite mínimo de cores encontrado.

O problema de emparelhamento em um grafo de *2n* vértices consiste em se escolher *n* arestas disjuntas, formando assim duplas ou pares de vértices, sendo que todos os vértices façam parte de exatamente um dupla, e cada dupla seja formada por vértices adjacentes no grafo. Sendo um grafo com pesos, podemos estar interessados em *matching* de peso mínimo ou máximo, ou seja, um emparelhamento dos vértices de tal forma que o peso total das arestas selecionadas seja o mínimo ou o máximo possível. Grafos com número ímpar de vértices não possuem um *matching* completo, pois nem todos os vértices poderão ser emparelhados. Isso ocorre mesmo em grafos com número par de vértices, dependendo da disposição das arestas. Em tais casos o problema de *matching* mínimo não será resolvido, e o de *matching* máximo deixará vértices isolados.

5. Exemplos de utilização

A seguir mostramos quatro exemplos de utilização do FEMG, todos exemplos típicos que podem ser dados em listas de exercícios de aplicação de grafos. Em cada exemplo é mostrado um problema e a forma de resolvê-lo utilizando a ferramenta. Vale ressaltar que utilizando a ferramenta o aluno pode manter o enfoque na modelagem e transformação de um problema real em um problema envolvendo grafos, recorrendo ao FEMG para encontrar a solução de sua modelagem, e assim realmente resolvendo o problema, analisando se o grafo retrata de forma correta o problema. Sem o uso da ferramenta, o aluno não teria a resposta completa do exercício, a não ser para grafos pequenos e tratáveis à mão, ou que implementasse um algoritmo, mas isso é um tarefa para disciplinas de programação.

5.1. Coloração de mapas

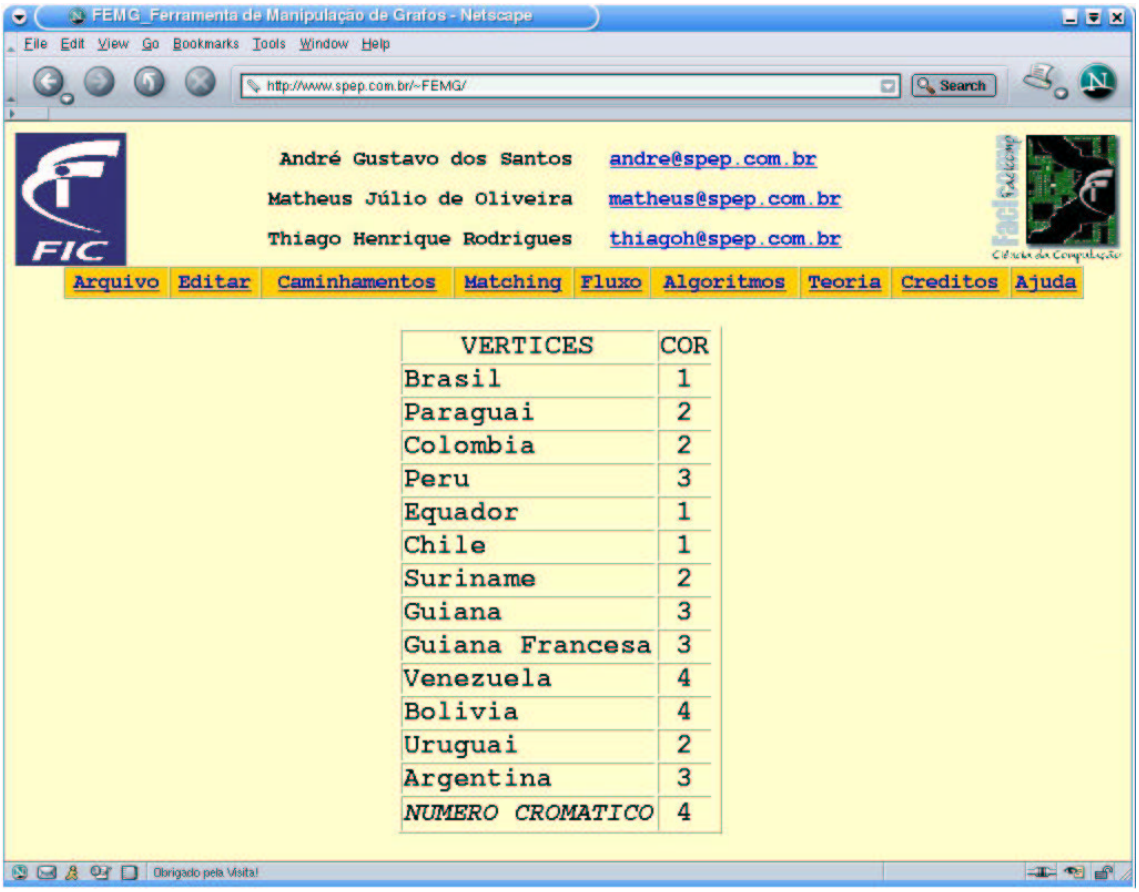
Queremos descobrir como colorir o mapa da América do Sul usando o número mínimo de cores, sem que dois países vizinhos sejam coloridos com a mesma cor. É conhecido que qualquer mapa pode ser colorido atendo a esta restrição, com no máximo 4 cores. Para descobrir qual cor usar em cada país construímos um grafo cujos vértices representam cada um dos 13 países, e cujas arestas conectam vértices cujos países representados tenham fronteira. Para obter a resposta do problema basta usar o algoritmo de coloração, cujo resultado é mostrado na figura 5.

5.2. Agendamento de horário de reuniões

Na semana anterior ao início das atividades do semestre, há reuniões das diversas comissões da faculdade: departamento financeiro, conselho acadêmico, diretoria e coordenadores, plenárias de curso com corpo docente e outras comissões de planejamento. Várias pessoas estão em mais de uma comissão, causando dificuldade no escalonamento de horários das reuniões, já que reuniões que contam com pelo menos uma pessoa em comum devem ser agendadas para horários distintos. Como decidir em que horário marcar cada reunião para realizar todas no número mínimo de horários?

Esse problema de escalonamento de horários pode ser resolvido através de coloração de grafos: cada comissão é representada por um vértice no grafo e há uma aresta no grafo para cada par de comissões que possuem pelo menos uma pessoa em comum. O número mínimo de cores representa o número mínimo de horários, cada cor

um horário, já que não haverá comissões (vértices) com pessoa em comum agendada para o mesmo horário (mesma cor).



VERTICES	COR
Brasil	1
Paraguai	2
Colombia	2
Peru	3
Equador	1
Chile	1
Suriname	2
Guiana	3
Guiana Francesa	3
Venezuela	4
Bolivia	4
Uruguai	2
Argentina	3
NUMERO CROMATICO	4

Figura 5: resultado do algoritmo de coloração para o mapa da América do Sul

5.3. Turismo

Suponha um viajante que deseja visitar em sua lancha 15 das ilhas da Polinésia Francesa, gastando o menor tempo nas viagens entre as ilhas, para aproveitar ao máximo o tempo descansando nas ilhas. Dada a localização geográfica das ilhas, pode-se montar um grafo completo onde o peso de cada aresta é o tempo de percurso entre as ilhas representadas pelos vértices incidentes. O problema do caixeiro viajante encontra o *tour* com o menor tempo total percorrido, como ilustrado no exemplo das figuras 3 e 4 para as capitais da região sudeste. Como o volume de dados é grande, um programa simples pode ser feito para gerar o arquivo de entrada.

5.4 Computadores em rede

Na instalação dos computadores de uma empresa, todos devem estar ligados em rede para permitir compartilhamento de recursos. Existem várias topologias propostas para conexão em rede, mas o interesse aqui será apenas que haja conexão direta ou indireta entre cada par de computadores, não importando o tráfego na rede e outros fatores, simplesmente que seja gasto o mínimo de instalação de infra-estrutura (cabos, mudanças nas paredes, etc). Dados os locais onde se quer instalar computadores, e o custo de ligação direta entre cada par de computadores, qual a instalação mais barata ?

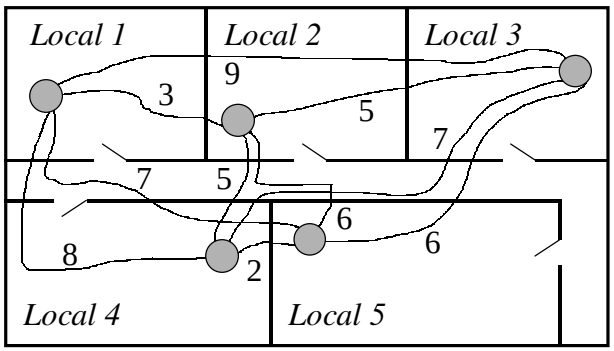
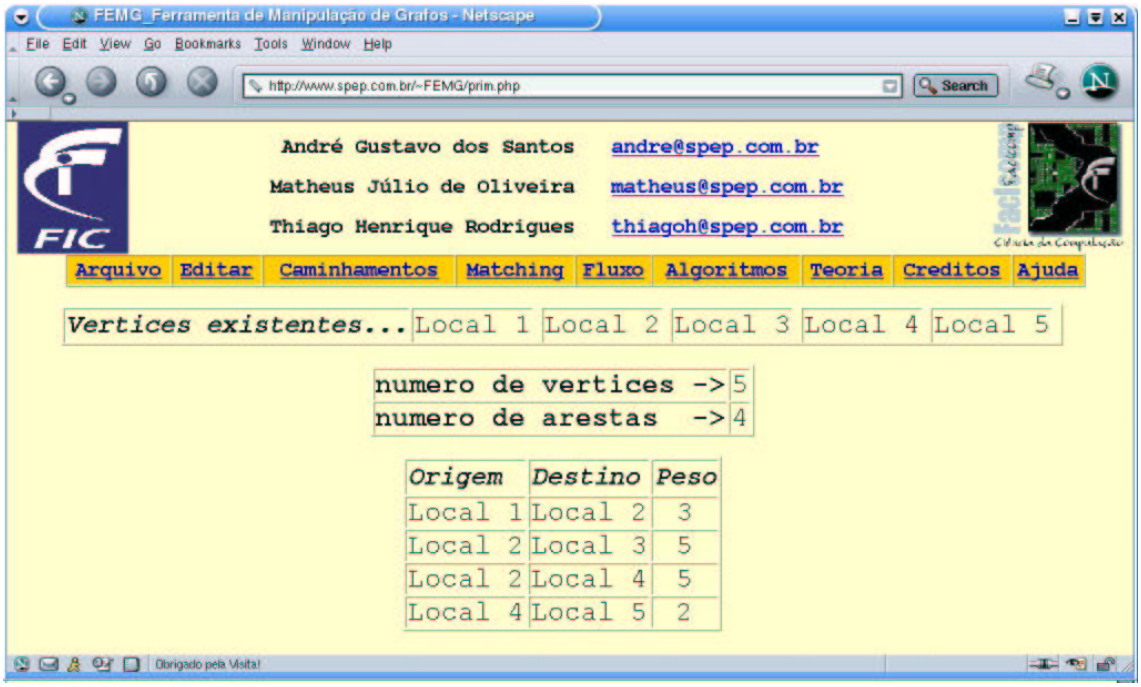


Figura 6: exemplo com 5 salas contendo o custo de cada ligação direta

Nesse caso montamos um grafo onde cada vértice é um local de instalação de computador, e cada aresta tem um peso de acordo com o custo da ligação direta. Se uma ligação direta for muito cara, ou mesmo inviável de ser feita, a aresta pode ser desconsiderada. Entrando os dados do grafo no FEMG, basta agora pedir uma árvore de custo mínimo, que teremos o total a ser gasto e quais as ligações a serem instaladas.



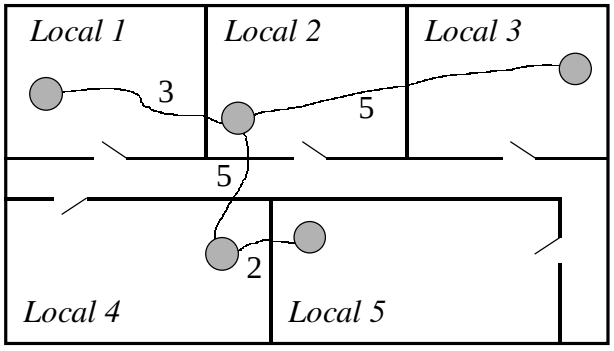


Figura 7: (a) resultado do algoritmo de árvore mínima para o problema de instalação da rede (b) representação do resultado

6. Conclusões

Outras funcionalidades estão sendo incorporadas ao FEMG, como desenho do grafo, testes de planaridade, e entrada de dados visualmente. FEMG está sendo avaliado através de sua utilização, na resolução de listas de exercícios e trabalhos, pelos alunos da faculdade matriculados nas disciplinas que possuem tópicos de teoria, algoritmos e aplicação de grafos como parte da ementa, como é o caso de Algoritmos e Estruturas de Dados e Matemática Discreta em nossa grade curricular. Em trabalhos sobre aplicação de grafos os alunos foram encorajados a não só modelar o problema mas encontrar a solução, seja escrevendo um programa, ou utilizando alguma ferramenta pronta. Vários deles utilizaram nossa ferramenta quando transformaram o problema em problemas de grafos envolvendo coloração, ciclo hamiltoniano, caminho mínimo e caixeiro viajante.

A ferramenta estará disponível no *site* da faculdade, através do endereço http://www.spep.com.br/~FEMG, podendo ser utilizada livremente por qualquer pessoa com acesso à internet. FEMG foi eficiente nos testes já realizados, com grafos de porte médio. Ainda estão sendo preparadas instâncias grandes, com dados reais, para testar seu desempenho. Entretanto, o motivo principal da criação do FEMG é sua utilização no meio acadêmico, onde são necessárias apenas instâncias pequenas ou de porte médio.

Enfim, além de servir como bom aprofundamento por parte dos autores no estudo de algoritmos e problemas envolvendo grafos, cremos estar disponibilizando uma ferramenta útil, prática, gratuita e acessível por qualquer aluno ou instituição, contribuindo para socializar o conhecimento e diminuir as diferenças entre o estudo em grandes universidades públicas e pequenas faculdades privadas, na área de ferramentas educativas e no ensino à distância.

6. Referências

Cormen, T.H. et al., Introduction to algorithms. 2. Boston: Mc Graw-Hill, 2001.

Dijkstra's Shortest Path Algorithm, Harvey J. Greenbeng, http://carbon.cudenver.edu/~hgreenbe/sessions/dijkstra/DijkstraApplet.html, fevereiro 2004

Evans, J.R., Minieka, E., Optimization Algorithms for Networks and Graphs, Marcel Dekker, 1992

GRIN - Graph Interface, Vitaly Pechenkin, http://www.geocities.com/pechv_r1/, janeiro 2004

Markenzon, L., Szwarcfiter, J.L., Estruturas de dados e seus algoritmos. 2. Rio de Janeiro: LTC, 1994

Ramalho, J.A., HTML dinâmico. São Paulo, Berkeley do Brasil, 1999

Thomson, L., Welling, L., PHP E MySQL, Desenvolvimento Web. Rio de Janeiro: Campus, 2001

Tucker, A., Applied Combinatorics.3. JohnWiley and Sons, 1994