

KITNET: UM PROTÓTIPO DE SOFTWARE DE VISUALIZAÇÃO DE REDES

Wagner Arbex¹

arbex@cnpqgl.embrapa.br

Lilian Markenzon²

markenzon@nce.ufrj.br

¹Embrapa Gado de Leite

R. Eugênio do Nascimento, 610 – 36038-330 – Juiz de Fora – MG

Universidade Presidente Antônio Carlos - UNIPAC

Centro de Ensino Superior de Juiz de Fora – CES/JF

²Núcleo de Computação Eletrônica – UFRJ

CP 2324 - 20001-970 - Rio de Janeiro – RJ

RESUMO

Softwares de visualização de redes enfrentam problemas específicos, tais como o tratamento de grandes volumes de informação, disponibilidade de traçados automáticos para facilitar o entendimento da rede, importação e exportação de arquivos, e tratamento de subredes. Neste trabalho apresentamos o KitNet, um protótipo de software de visualização de redes (SVR), que trata dois dos problemas comuns a SVR: o desenho de redes por meio de traçados automáticos e a importação e exportação de arquivos descritores da rede. Além disso, gera redes simuladas para serem utilizadas em testes ou como exemplos.

Palavras-chave: Visualização de redes, Traçado automático, Software de visualização.

ABSTRACT

Network visualization deals with typical problems such as huge volumes of information, the need of automatic drawing tools in order to provide a better understanding, the manipulation of files and the treatment of subnets. In this paper we present the KitNet, a prototype package for network visualization that implements an integrated solution for automatic drawing of nets and files manipulation. Furthermore, it generates simulated nets to be used in tests or as examples.

Key-words: Network visualization, Graph drawing, Visualization software.

1 Introdução

O Novo Dicionário Aurélio (FERREIRA, 1999) define visualização como a transformação de conceitos abstratos em imagens real ou mentalmente visíveis. Por sua vez, "visualizar" significa formar ou conceber uma imagem visual mental de algo que não se tem ante

os olhos no momento ou, ainda, tornar visível mediante manobra ou procedimento.

Contextualizando o termo para uso específico em sistemas de computação, Price et al. (1993), o definem como a capacidade ou processo de formação de uma imagem mental ou visão de algo que não está sendo visto no momento, podendo resultar da percepção de qualquer dos sentidos humanos. É interessante ressaltar que esta nova definição, apesar de ser voltada para o contexto da informática, apenas reforça o conceito anterior, uma vez que não traz uma nova perspectiva ou especificidade. Assim sendo, a abordagem desta introdução é discutir tais conceitos, entre outros, e situá-los neste contexto mais específico.

Os termos “visualização” e “animação” representam conceitos diferentes. Entretanto, se confundem por serem inúmeras vezes utilizados em conjunto, por possuírem técnicas comuns ou por serem complementares.

Por Domingue (1995), animação de algoritmos é a caracterização em alto nível de como os dados são manipulados durante a execução dos programas que os implementam ou, ainda, por Brown (1988), animação de algoritmos mostra uma abstração em alto nível do código e dos dados de um programa, criando uma representação visual dos elementos de um programa, isto é, dos algoritmos e das estruturas de dados. Na animação transfere-se para a imagem algum tipo de entendimento de fenômenos dinâmicos, buscando “dar vida a alguma coisa” (SORENSEN, 2001).

Tais definições podem ser adotadas quase totalmente para visualização. Contudo, pode-se desprezar o dinamismo que a animação traz consigo, pois o objetivo final da visualização não é “dar vida”, mas permitir representações visuais do objeto de estudo.

A implementação de tais conceitos é feita por meio dos softwares de visualização (SV), que são ferramentas desenvolvidas com uso de técnicas de cinema, projeto gráfico e de desenho animado para apresentar estruturas de dados, programas ou algoritmos (PRICE et al., 1993). De acordo com Domingue (1995), SV é basicamente a unificação de animação de algoritmos e visualização de programas, conceitos desenvolvidos nos anos 80.

Um problema específico de visualização é a representação de redes de serviço, ou simplesmente redes, definidas como $N = (G, I)$, onde G é grafo $G = (V, E)$, V o conjunto de nós,

E o conjunto de arestas de G , e I o conjunto de informações sobre os nós e arestas de G .

Redes são modelos matemáticos construídos para a representação de sistemas físicos ou abstratos (Coullard et al., 1999), sendo necessária a apresentação da estrutura, da organização e do funcionamento destes, em que os nós podem representar, segundo He (1999), uma entidade “física” (uma cidade, um ponto de fornecimento de energia, um site etc.) ou “não-física” (um IP, um domínio, um endereço de uma home page etc.). A ligação entre os nós pode representar um link de comunicação, um cabeamento telefônico ou elétrico etc., dependendo do sistema ou serviço representado, e, assim como os nós, cada qual possui suas próprias características e informações. Estas, por sua vez, se dividem entre “permanentes”, como a capacidade nominal máxima da ligação, e “temporárias”, como o fluxo na ligação em um determinado instante. Essas informações são valores agregados à entidade, não a representando em si, que, como dito, pode nem mesmo existir fisicamente.

Este artigo apresenta o KitNet, um protótipo de software de visualização de redes (SVR), que trata dois dos problemas comuns a SVR: o desenho de redes por meio de traçados automáticos e a importação e exportação de arquivos descritores da rede. Além disso, gera redes simuladas para serem utilizadas em testes ou como exemplos. Detalhes podem ser vistos em Arbex (2002).

2 Levantamento de problemas em visualização de redes

Trabalhos em visualização são encontrados desde o início da década de 80 em diversas aplicações, e na última década percebe-se o crescente interesse na área, especificamente em visualização de redes.

Os SVR, por natureza da aplicação, enfrentam alguns problemas típicos, tais como contextualização da rede representada para o universo de aplicação do usuário, tratamento e representação de grandes volumes de informações, disponibilidade de traçados para facilitar a leitura e o entendimento da rede, interface entre softwares para comunicação entre estes, e tratamento e manipulação de blocos ou subredes.

Becker et al. (1990) abordam os problemas de representação de grande volume de dados

e contextualização da representação. Defendem que caso fosse utilizada uma representação (p. ex., cores com significados definidos para os nós e ligações), e se a rede fosse disposta visualmente sobre seu contexto (p. ex., uma malha rodoviária visualizada sobre um mapa), facilitaria a leitura e o entendimento da rede. O problema nesta proposta, segundo os autores, é que possivelmente tem-se “um pequeno número de linhas e símbolos contrapondo-se a um grande volume de dados, o que pode tornar-se ilegível”. Todavia, apresentam a solução por meio de duas técnicas gráficas dinâmicas para visualização de dados de redes. A primeira tratando os dados das ligações e a outra os dados dos nós, ambas com o objetivo de “limpar” a visualização da rede.

O artigo de Eick & Wills (1993) apresenta um SVR que trata a representação de grandes volumes de informações, a contextualização, o tratamento e a manipulação de blocos e, indiretamente, o traçado, por meio o posicionamento dos nós.

Os autores contextualizam redes de sistemas de computação utilizando cores com significado conhecido para informações relativas aos nós e ligações, assim como descrevem e utilizam um método de posicionamento dos nós no plano a partir da associação de “força” e “peso” às ligações, fazendo com que nós fortemente ligados fiquem próximos e, desta forma conferindo um significado relativo ao posicionamento.

O tratamento de grandes volumes de informações é feito pela da interpretação da hierarquia, que representa o agrupamento de objetos relacionados. Por exemplo, se representarmos graficamente o desenvolvimento de um software como uma rede, temos os arquivos fontes, que podem ser estruturados em módulos, e estes formam subsistemas.

Becker et al. (1995) apresentam o SeeNet, um SVR que trata dados associados a redes de grandes dimensões, o que invariavelmente gera grande volume de dados, e enfatizam a abundância de informações possíveis de serem obtidas. Apontam a visualização como uma grande estratégia para o entendimento dessas e, para tanto, o SeeNet permite três diferentes vistas de uma rede: duas relacionadas à disposição geográfica da rede, buscando a contextualização, e a outra uma disposição matricial. Todas podem ser parametrizadas para análise do comportamento da rede.

Outro SVR que também merece destaque é o Graphical Implementation Development

Environment for Networks, GIDEN, desenvolvido por Coullard et al. (1999). O GIDEN é um ambiente de software interativo projetado para facilitar a visualização de problemas, de soluções e de algoritmos de otimização de redes.

Com exceção do GIDEN, todos os demais SVR citados abordam com maior ou menor ênfase a representação de grandes volumes de informações, e tal problema se agrava ainda mais se for considerado o crescimento quantitativo das informações disponíveis, bem como a crescente variação das formas em que podem se apresentar.

Sob a obtenção de muitas informações escondem-se, na verdade, dois problemas: a filtragem, ou seleção, da informação a ser apresentada, e a representação da informação. Neste ponto merecem destaque os trabalhos de Carmo & Cunha (1998) e Alves et al. (2000), que descrevem modelos de visualização, partindo de técnicas de filtragem, seleção e de representação.

Alves et al. (2000) apresentam o SV VisWeb, uma ferramenta de visualização totalmente voltada para aplicações web originada a partir das técnicas de abordagem dos problemas de visualização. Especificamente para grandes volumes de informações, os autores apresentam a proposta de dividir em etapas distintas (“pipeline”) o processo de geração da imagem que será visualizada, sendo composto por uma série de procedimentos com fins específicos: filtragem, mapeamento e “rendering”, para, então, nesta ordem, produzirem a representação visual esperada. Esse processo é similar ao apresentado por CARMO & CUNHA (1998).

Em He (1999) é apresentado um visualizador para um simulador de rede de comunicação de dados proprietário desenvolvido pela Lucent Technologies. Uma ferramenta como essa permite simular as operações e o estado de uma rede, sendo útil a projetistas e gerentes, pois facilita a avaliação do comportamento da rede em diversos cenários por meio da alteração de seus parâmetros, tais como topologia, roteamento e tráfego, sendo possível representar uma rede existente, simular diferentes tipos de falhas em nós e ligações e observar seus impactos, analisar o efeito do controle de gerenciamento da rede e otimizar seu projeto. Para um simulador é de crucial importância o desenvolvimento de visualizadores.

3 O KitNet

Na seção anterior se observa que, dentre os problemas citados, a grande preocupação dos autores está relacionada com o tratamento e representação de grandes volumes de informações e, em seguida, com a contextualização da rede representada para o universo de aplicação do usuário. Os demais problemas ou não são tratados, talvez porque a aplicação não exija, ou são tratados em questões particulares da aplicação, ou, ainda, são tratados indiretamente, como efeito colateral.

O KitNet atua sobre o traçado, o desenho e a apresentação de uma rede, permite a representação e manipulação visual de um sistema e implementa um conjunto mínimo de funcionalidades necessárias em um SV.

Apesar de ser um protótipo, o KitNet apresenta grande facilidade de operação, que não se pretende modificar em versões futuras, e disponibiliza uma opção de criação da rede de forma aleatória e automática, com a intenção de permitir a visualização de um sistema, mesmo que um usuário praticamente não saiba como criá-lo. Esta opção torna-se muito útil à medida que o usuário pode utilizá-la como um ensaio e para aprender como manipular as características do software, além de ser um ótimo recurso para gerar redes para testes.

O KitNet trata duas frentes: permite a integração e comunicação com outros softwares, pela importação e exportação dos arquivos de informações das redes, disponibiliza traçados automáticos, e ainda oferece algoritmos básicos de buscas (largura e profundidade).

Ressalta-se o grande benefício que se obtém, com um custo relativamente baixo, com a disponibilização de diferentes traçados para o usuário, visando facilitar a leitura de uma rede de serviços. Deve-se ressaltar também que mesmo trabalhos mais sofisticados em visualização de rede, tais como He e Eick (1998) e He (1999), não se preocupam em desenhar o sistema de outras formas, mas buscam facilitar o entendimento através de outras técnicas, que podem ser aplicadas em conjunto ou isoladamente, tais como filtros, uso de símbolos e convenções para representação, ou a apresentação da rede sobre um contexto visual amigável.

4 Traçados Automáticos

A proposta de utilizar diferentes traçados para facilitar a leitura e entendimento da rede representada, no âmbito da discussão, somente foi considerada e constatada por Eick & Wilks (1993) quando afirmam que, apesar de não intencionalmente, foi obtida uma harmonia no desenho, a partir do estabelecimento de uma relação para o posicionamento dos nós.

A visualização de uma rede no KitNet é implementada disponibilizando traçados automáticos para o usuário, oferecendo a possibilidade de enxergar o sistema representado por diferentes desenhos, que podem representar diferentes topologias, e fornecendo novas perspectivas para a leitura da rede. O protótipo disponibiliza quatro opções de traçados: o traçado aleatório, o traçado em coroa, o traçado segundo o algoritmo de Kamada e Kawai (KAMADA & KAWAI, 1989) e o traçado em níveis.

O traçado aleatório simplesmente sorteia os nós da rede dentro da área de exibição, tomando o cuidado para não coincidir dois nós em uma mesma posição e estabelecendo as propriedades dos nós e ligações, que são mantidas nos demais traçados.

Para a implementação do traçado em coroa, calculam-se as coordenadas polares, X e Y , para os nós e os distribui sobre uma circunferência de raio R , com os respectivos valores sendo obtidos conforme mostrado a seguir. NP é o número de nós da rede, H e L são, respectivamente, a altura e a largura da janela de exibição e C uma constante para a obtenção de um raio bem dimensionado para o tamanho da janela de exibição.

$$R = \frac{C \cdot NP}{2}; R \leq \frac{H}{2} \quad X = R \cdot \cos\phi + \frac{L}{2}; \frac{2\pi}{NP} \leq \phi \leq 2\pi \quad Y = R \cdot \sin\phi + \frac{H}{2}; \frac{2\pi}{NP} \leq \phi \leq 2\pi$$

O traçado pelo algoritmo de Kamada e Kawai foi desenvolvido em grande parte pelo porte da implementação de Vernet (1993). Por Quaresma e Lopes (1991), o algoritmo de Kamada e Kawai enxerga cada par de nós como ligados por uma determinada força e busca atingir um estado de energia mínima no sistema como todo, pelo somatório das forças, e objetiva satisfazer critérios estéticos, tais como equilíbrio, densidade, simetria e isomorfismo.

A energia total (E) do sistema é dada por: $E = \sum_{i=1}^{n-1} \sum_{j=i+1}^n \frac{1}{2} k_{ij} (|p_i - p_j| - l_{ij})^2$, sendo p_i ($1 \leq i \leq n$) as “partículas de energia” (distância) entre os nós n_1, \dots, n_n , l_{ij} o comprimento desejado entre dois nós, calculado a partir do caminho mais curto entre n_i e n_j , e k_{ij} a força existente entre n_i e n_j , que é inversamente proporcional ao caminho mais curto entre os nós. O objetivo do algoritmo é minimizar E , pela da resolução iterativa de um sistema de equações baseado no método de Newton-Raphson.

No desenvolvimento deste traçado foram detectadas algumas condições restritivas para a implementação deste algoritmo. O algoritmo não trata redes desconexas e é bastante comum não convergir para redes com um número maior de ligações ou cujos nós estejam de alguma forma alinhados. Além disso, Quaresma e Lopes (1991) afirmam que as constantes utilizadas no algoritmo devem ser diferentes de grafo para grafo e de desenho para desenho, sob pena do algoritmo não atingir uma posição estável, ou seja, não convergir, sendo necessário o desenvolvimento de um detector de ciclos para esses casos.

O quarto e último desenho de traçado disponibilizado ao usuário é o traçado em níveis, que cria e representa visualmente uma relação hierárquica na rede. Essa hierarquia é obtida por meio da árvore de caminhos gerada por busca em largura, iniciada em um nó (raiz) origem da busca. A busca em largura foi implementada segundo o algoritmo proposto por Szwarcfiter (1986), onde, durante a busca, é determinada a ordem dos nós visitados, dada pela largura dos nós e, como consequência, são obtidas as ligações do caminho percorrido (arestas da árvore de largura). As demais ligações foram chamadas de ligações alternativas. Foi adotada a convenção de linha cheia para as ligações do caminho percorrido e tracejada para as ligações alternativas.

Para o traçado em níveis, são disponibilizados dois desenhos: com posicionamento dos nós em linha reta ou com posicionamento radial. No primeiro caso, os nós são distribuídos, em cada nível, sobre uma linha imaginária fixada a uma distância $Y_i = DN \times (i-1) + C$ da raiz e mantendo uma distância $DN = \frac{H}{n}$ do nível anterior. Considerando H a altura da janela de exibição, n o maior nível da árvore de busca, i o nível corrente e C uma constante utilizada para obter uma distância bem dimensionada para o tamanho da janela de exibição.

Habilitando a opção de posicionamento radial, os nós são distribuídos em

semicircunferências concêntricas com centro na raiz, sendo uma semicircunferência para cada nível com raio $Y_i = |\text{sen}\phi| \times DN \times (i-1) + C$; $0 \leq \phi \leq \pi$, mantendo-se as definições das variáveis e constantes.

Tanto para o posicionamento reto dos nós, quanto no posicionamento radial, a coordenada X é obtida por: $X_i = \cos\phi \times DN \times (i-1) + \frac{L}{2}$; $0 \leq \phi \leq \pi$, também mantendo-se todas as definições anteriores e L representa a largura da janela de exibição. No posicionamento radial as coordenadas X_i e Y_i são polares e geram o plano de Argand-Gauss, assim como no traçado em coroa.

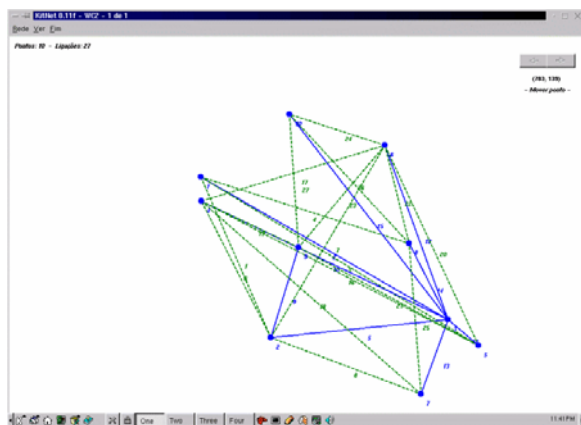


Figura 1: Rede criada a partir da importação do arquivo apresentado na Figura 5, com 10 nós dispostos aleatoriamente e 27 ligações. Exibição sem grade ou moldura. Apresenta busca em largura, partindo do nó 3.

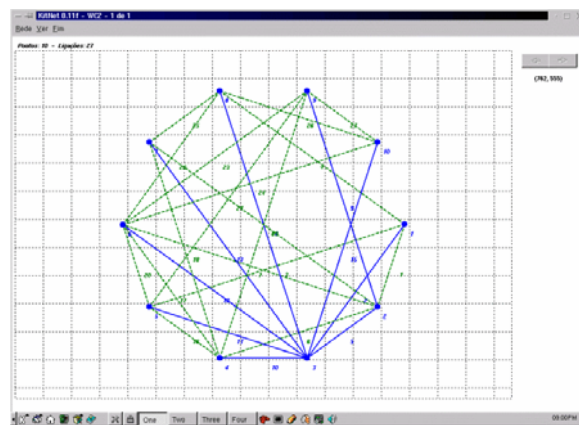


Figura 2: Traçado em coroa sobre a rede apresentada na Figura 1. Exibição com grade e moldura.

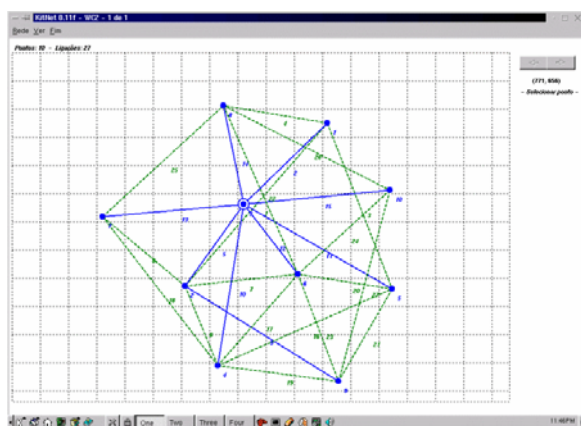


Figura 3: Traçado pelo algoritmo de Kamada e Kawai sobre a rede apresentada na Figura 1. Exibição com grade e moldura.

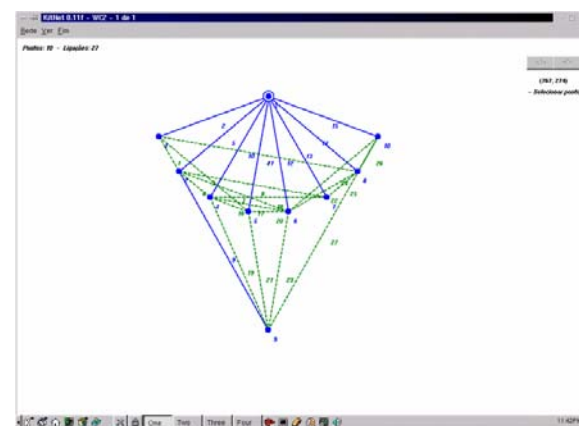


Figura 4: Traçado em níveis, com posicionamento radial dos nós, sobre a rede apresentada na Figura 1. Exibição sem grade ou moldura.

5 Importação e exportação de redes

Além de tratar a representação visual de redes, o KitNet oferece um conjunto de procedimentos para que o usuário seja capaz de criar, editar, alterar, armazenar e ler uma rede. No conjunto de procedimentos de leitura e armazenamento, existe o recurso de importação e exportação que permite a troca de dados entre softwares, e a rotina de importação possibilita ler arquivos textos em formato específico ou no formato KineGraph.

O KineGraph, atualmente na versão 3.0, é um ambiente para a implementação de algoritmos em grafos, sem, contudo, disponibilizar um traçador (visualizador) de grafos, visto não ser este seu objetivo. Segundo Markenzon & Vernet (1998), o KineGraph é um conjunto de ferramentas, programas e tipos abstratos de dados, disponíveis como bibliotecas de rotinas, que permite o projeto, a implementação e o acompanhamento (“debugging”) de algoritmos em grafos de forma simples e integrada.

Os arquivos do tipo texto para importação devem atender a uma pequena especificação de formato, que será explicado a partir do exemplo mostrado na Figura 5. O exemplo está com as linhas numeradas apenas para ser usado como referência no decorrer deste texto, sendo este o arquivo de dados que originou a rede exibida nas Figuras 1 a 4.

```
(1) 10  
(2) 1 2 3 5 8  
(3) 2 1 3 4 6 7 9  
(4) 3 1 2 4 5 6 7 8 10  
(5) 2  
(6) 4 2 3 5 6 7 9  
(7) 5 1 3 4 6 9  
(8) 6 2 3 4 5 8 9 10  
(9)  
(10)7 2 3 4 8  
(11)8 1 3 6 7 10  
(12)9 2 4 5 6 10  
(13)15 *  
(14)10 3 6 8 9
```

Figura 5: Exemplo de arquivo texto para importação.

Um arquivo texto para importação parte das seguintes regras:

- 1) A primeira linha deve conter somente a quantidade de nós da rede.
- 2) O primeiro campo de cada linha identifica o nó cujas informações constam nesta mesma linha, caracterizando o nó corrente. Todas as demais informações serão os outros nós que estão ligados ao nó corrente ou às coordenadas de localização do nó corrente. Se não existir a informação das coordenadas, a localização do nó corrente será aleatória.
- 3) As coordenadas são identificadas pelos caracteres “c” ou “C”, seguidos de dois valores: coordenadas X e Y. Caso não exista valor para X ou Y, ou estiver fora dos limites das janelas de exibição, então o valor será escolhido aleatoriamente.
- 4) A presença de um asterisco em uma linha indica que o nó corrente possui ligação para todos os nós da rede, exceto para si próprio.
- 5) Ao iniciar a leitura de uma linha, o protótipo ignora qualquer caracter até que surja um primeiro algarismo e, a partir deste, identifica o número iniciado por este algarismo.
- 6) Os campos em uma linha são separados pelo caracter branco. Na realidade, com exceção dos caracteres “c” e “C”, qualquer caracter diferente de algarismo é considerado separador de campo.
- 7) Um mesmo nó pode ter suas informações divididas em duas ou mais linhas, bastando que as linhas iniciem com a sua identificação.
- 8) Informações duplicadas são ignoradas, mas informações redundantes, se possuírem algum significado lógico, substituem a anterior. Por exemplo, se forem informadas duas ou mais coordenadas para um mesmo nó, será mantida a informação mais recente.

De acordo as regras especificadas, podem ser feitas algumas observações a respeito do arquivo apresentado na Figura 5, por exemplo: a linha “1” informa que a rede possui 10 nós; a linha “2” informa que existem as ligações (1, 2), (1, 3), (1, 5) e (1, 8); a linha “9” será ignorada pois está em branco; a linha “13” informa que o nó corrente está ligado a todos os demais nós, entretanto será ignorada, pois não existe o nó corrente (nó 13) em uma rede com 10 nós.

A rotina de exportação faz o trabalho inverso gerando arquivos no formato texto já

descrito. Com a opção de o arquivo ser composto somente com a identificação nós da rede e as respectivas ligações ou, ainda, com as coordenadas dos nós. Esses dois casos podem ser vistos na Figura 6, onde foi exportada rede da Figura 4.

10	10
1 2 3 5 8	1 2 3 5 8 c 251 174
2 1 3 4 6 7 9	2 1 3 4 6 7 9 c 287 236
3 1 2 4 5 6 7 8 10	3 1 2 4 5 6 7 8 10 c 446 103
4 2 3 5 6 7 9	4 2 3 5 6 7 9 c 342 282
5 1 3 4 6 9	5 1 3 4 6 9 c 410 307
6 2 3 4 5 8 9 10	6 2 3 4 5 8 9 10 c 481 307
7 2 3 4 8	7 2 3 4 8 c 549 282
8 1 3 6 7 10	8 1 3 6 7 10 c 604 236
9 2 4 5 6 10	9 2 4 5 6 10 c 445 517
10 3 6 8 9	10 3 6 8 9 c 640 174

Figura 6(a): Arquivo de exportação da figura 4 sem as coordenadas dos nós

Figura 6(b): Arquivo de exportação da figura 4 com as coordenadas dos nós

6 Conclusões

Ao se desenvolver uma ferramenta de visualização, deve-se estar ciente de que a visualização é uma área dependente da tecnologia de recursos computacionais, de plataformas de hardware, de linguagens e técnicas de programação, e facilmente muitos trabalhos tendem a tornar-se obsoletos em um curto espaço de tempo. Além disso, é natural que novos recursos computacionais permitam a expansão ou ampliação dos trabalhos nesta área. Atualmente as ferramentas de visualização estão voltadas para plataformas web, com muitos recursos de visualização interativa e utilização de arquitetura cliente/servidor.

Para este trabalho, a plataforma computacional utilizada foi um microcomputador dotado de processador Intel Pentium II, frequência de 400 MHz, com 128 MB de memória, tendo como sistema operacional o GNU/Linux, kernel 2.2.16-22, distribuição Red Hat versão 7.0. Como ambiente de desenvolvimento foi utilizado o Kylix 1.0, versão Open Edition. Ao final o protótipo foi migrado para o ambiente Kylix 2.0, também na versão Open Edition.

Como ampliação e complemento deste trabalho sugere-se combinar as técnicas de

traçados com a contextualização para o universo de conhecimento do usuário e implementar o tratamento de blocos, fundamental para representação de grandes volumes de informações.

7 Referências Bibliográficas

ALVES, A. D., OLIVEIRA, M. C. F. de, NONATO, L. G. **Interactive Visualization over the WWW**. In: Brazilian Symposium on Computer Graphics and Image Processing – SIBGRAPI 2000, 13., Gramado. Anais... Los Alamitos: IEEE Computer Society, 2000. p. 259 - 266.

ARBEX, W. **Aspectos de Visualização de Redes**. Dissertação (Mestrado em Sistemas e Computação) – Departamento de Engenharia de Sistemas, IME, Rio de Janeiro, 2002.

BECKER, R. A., EICK, S. G., MILLER, E. O., WILKS, A. R. **Network Visualization**. In: International Symposium on Spatial Data Handling, 4., 1990, Zurich. Proceedings... v. 1, 1990, p. 285 - 294.

BECKER, R. A., EICK, S. G., WILKS, A. R. **Visualization Network Data**. IEEE Transactions on Visualization and Computer Graphics, v. 1, n. 1, march 1995, p. 16 – 28.

BROWN, M. **Algorithm Animation**. An ACM Distinguished Dissertation 1987. New York: MIT Press, 1988.

CARMO, M. B., CUNHA, J. D. **Filtragem e Representação na Visualização de Informação**. In: Encontro Português de Computação Gráfica, 8., 1998. Anais eletrônicos... Disponível em: <http://automatix.inesc.pt/>. Acessado em 15 de Março de 2001.

COULLARD, C. R., DILWORTH, D. S., OWEN, J. H. GIDEN. **A Graphical Environment for Network Optimization** - Version J-2.0 Beta. User Guide. November, 1999.

DOMINGUE, J. **Using Software Visualization Technology in the Validation of Knowledge Based Systems**. In: Workshop on Knowledge Acquisition for Knowledge-Based Systems, 9., 1995, Banff, Canadian. Proceedings... Banff, Canadian, 1995. Disponível em: <http://kmi.open.ac.uk/people/domingue/pubs/pubs.html>. Acessado em 15 de Fevereiro de 2001.

EICK, S. G., WILLS, G. J. **Navigating Large Networks with Hierarchies**. In: IEEE Conference on Visualization. Proceedings... 1993, p. 204 - 210.

FERREIRA, A. B. de H. **Novo Dicionário Aurélio**. Rio de Janeiro: Nova Fronteira, 1999.
HE, T. Internet-Based Front-End to Network Simulator. In: Data Visualization 1999. Proceedings of Eurographics 99. May 1999. p. 247 – 252.

HE, T., EICK, S. G. **Constructing Interactive Network Visual Interfaces**. Disponível em: <http://www.bell-labs.com/user/taosong>. Acessado em 15 de Janeiro de 2001.

KAMADA, T., KAWAI, S. **An Algorithm for Drawing General Undirected Graphs**. Information Processing Letters, v. 31, 1989, p. 7-15.

MARKENZON, L., VERNET, O. **Kinegraph: Manual do Usuário - Versão 3.0**. 1997.

PRICE, B. A., BACKERT, R. M., SMALL, I. S. **A Principled Taxonomy of Software Visualization.** Journal of Languages and Computing, v. 4, 1993, p. 211 – 266.

QUARESMA, P., LOPES, J. G. P. **Automatização do Desenho de Grafos e sua Aplicação em Inteligência Artificial.** Disponível em: <http://www.di.uevora.pt/~pq/papers/sbia91.pdf>. Acessado em 11 de Maio de 2002.

SORENSEN, V. **Missionária da Animação.** Ciência Hoje, Rio de Janeiro, v. 29, n. 174, Ago. 2001. p. 6 – 9.

SZWARCFITER, J. L. **Grafos e Algoritmos Computacionais.** 2 ed. Campus, Rio de Janeiro, 1986.

VERNET, O. **Kamada.** Rio de Janeiro, RJ, 1993. 1 disquete; 3 ½ pol. Programa aplicativo.