SPOLM 2008

ISSN 2175-6295 Rio de Janeiro- Brasil, 05 e 06 de agosto de 2008.

PROBLEMA DO CAMINHO MAIS CURTO – ALGORITMO DE DIJKSTRA

Yasmín Salazar Méndez
Mestrado em Engenharia de Produção – Universidade Federal Fluminense
Rua Passo da Pátria 156, São Domingos, CEP 24210-240, Niterói, RJ
jazmin5_79@hotmail.com

Luis Ernesto Torres Guardia
Departamento de Engenharia de Produção – Universidade Federal Fluminense
Rua Passo da Pátria 156, São Domingos, CEP 24210-240, Niterói, RJ
tepletg@vm.uff.br

Resumo

Este artigo trata sobre o problema do caminho mais curto na teoria e na prática. Na parte teórica apresenta-se a formulação matemática do problema, uma descrição geral de algumas aplicações e pesquisas relacionadas com este tema e os algoritmos do caminho mais curto do método de rotulação. Na parte prática apresenta-se a descrição do algoritmo de Dijkstra assim como a determinação do caminho mais curto entre dois nós específicos de duas redes de pequeno porte. Para a resolução destas aplicações o algoritmo de Dijkstra foi implementado.

Palavras-chave: Grafos, Redes, Problema do caminho mais curto.

Abstract

This paper presents the shortest path problem in theory and practice. The theoretical part contains the mathematical formulation of the problem, describes some applications in general and researches of the topic and presents the labeling shortest path algorithms. The practical part contains the description of the Dijkstra's algorithm and the determination the shortest path between two specified nodes of two networks of small size. To solve these applications, the Dijkstra's algorithm was implemented.

Key-word: Graphs, Networks, Shortest path problem.

1. INTRODUÇÃO

O problema do caminho mais curto "*Shortest path problem*"[1], consiste em encontrar o melhor caminho entre dois pontos chamados nós. Assim, resolver este problema pode

[1] Nome com que é conhecido o problema do caminho mais curto na literatura americana.

significar determinar o caminho entre dois nós com o custo mínimo, ou com o menor tempo de viagem ou com a máxima capacidade, (Ahuja et al., 1993).

O problema do caminho mais curto constitui o maior grupo na área de pesquisa operacional e é considerado como um dos mais importantes da programação linear (Hillier e Lieberman, 1988).

Glover et al. (1985), consideram o caminho mais curto como uma importante área de pesquisa devido à grande quantidade de aplicações práticas.

As aplicações do problema do caminho mais curto estão relacionadas com a otimização de certas atividades como, por exemplo: o tráfego de estradas, linhas de transmissão elétrica, conexão de redes, problemas de programação de rota crítica PERT, planejamento de movimentos de um robô e outras mais complexas como no campo de biologia molecular, (Eppstein, 1994).

Ahuja et al. (1993), citam mais outras aplicações dos algoritmos para resolver o problema do caminho mais curto e estão relacionadas com a realização ótima de planos de substituição de equipamento, elaboração de projetos, gestão de fluxo de caixa, transmissão de mensagens em sistemas de comunicação e fluxo de tráfego em cidades com muita congestão.

Glover et al. (1985), mencionam as seguintes aplicações práticas: substituição de equipamento, planejamento de rotas e tempo de viagens, roteamento e programação de veículos, planejamento de capacidade, desenho e/ou expansão de redes de transporte e comunicação, e programação de caminhos críticos.

Além das aplicações que estão associadas diretamente à busca do caminho mais curto, alguns outros problemas de programação inteira podem ser resolvidos através da formulação deste problema em forma direta ou aplicando alguma estratégia de relaxamento.

Glover et al. (1985) citam os seguintes problemas de programação inteira que foram resolvidos com o critério do caminho mais curto: localização, problema generalizado de atribuição, problema de emparelhamento máximo, problema do caixeiro viajante, problema da mochila e problemas de equilíbrio de tráfego.

Hung e Divoky (1990) mencionam que os algoritmos de busca do caminho mínimo têm sido utilizados para resolver problemas de emparelhamento, problemas de fluxo de custo mínimo e problemas de atribuição.

As diferentes situações que na prática requerem encontrar o caminho mais curto têm feito com que este problema tenha sido dividido em diferentes casos, a fim de facilitar seu estudo, pesquisas e desenvolvimento. Dreyfus (1969) apresenta cinco tipos de problemas de busca do caminho mais curto:

- Determinar o caminho mais curto entre dois nós específicos de uma rede.
- Determinar o caminho mais curto entre todos os pares de nós da rede.
- Determinar o segundo, terceiro, quarto, etc., caminho mais curto.
- Encontrar o caminho mais rápido numa rede com tempos de viagem e dependendo de uma hora de saída.
- Encontrar o caminho mais curto entre dois nós específicos precisando-se atravessar obrigatoriamente determinados nós intermediários.

O problema do caminho mais curto é um tópico atrativo para pesquisadores e profissionais, pois sua resolução permite obter soluções eficientes a importantes e freqüentes problemas reais ao determinar de modo confiável a forma mais rápida e econômica de realizar uma atividade determinada. Outro ponto que faz com que o tema seja atrativo para pesquisadores é porque tem a característica de que a partir de modelos de redes simples podem-se elaborar modelos mais complexos que são estudados na área de otimização combinatória. Não obstante que a solução de problemas do caminho mais curto seja relativamente fácil, o desenho e a análise de algoritmos eficientes de solução exigem muito engenho, (Ahuja et al., 1993).

O interesse que existe por realizar pesquisas relacionadas com o caminho mais curto não é recente, assim como o detalharam Dreyfus (1969) e Schrijver (2005), nas resenhas históricas que realizaram sobre este tema.

Encontrar o caminho mais curto num esquema de labirinto deu início aos problemas clássicos de grafos, destacando-se neste âmbito autores como Wiener (1873), Lucas (1882), e Tarry (1895), (Biggs, Lloyd e Wilson[2] 1976, apud Schrijver, 2005).

Especificamente os problemas do caminho mais curto somente foram estudados a partir da década dos 50 com a idéia de achar uma rota alternativa no caso de que uma rota estiver bloqueada. Trueblood (1952) desenvolveu um algoritmo para achar o caminho mais curto aplicado ao roteamento de ligações telefônicas.

É importante mencionar, que durante algum bom tempo os métodos heurísticos cobraram importância e foram objeto de intensa pesquisa, pois para resolver muitos problemas como por exemplo para determinar a melhor rota para viajar entre dois cidades por estrada, utilizaram-se heurísticas, (Rosenfeld[3], 1956 apud Schrijver 2005).

De 1946 – 1953 foram desenvolvidos métodos de matrizes para determinar o caminho mais curto, tendo como objetivo identificar num grafo dirigido os nós mais próximos. Estes métodos têm aplicações em redes neurais e em sociologia animal. Dentro destes estudos destacaram-se: Landahl e Runge (1946), Landahl (1947), Luce e Perry (1949), Luce (1950), Lunts (1950, 1952) e Shimbel (1953).

Para achar o caminho mais curto para ir de um nó específico até outro nó num grafo dirigido, surgiram dois tipos de métodos, um deles trabalha com redes com pesos arbitrários e o outro trabalha com redes com pesos não negativos. Shimbel (1955), Bellman (1958) e Moore (1959), desenvolveram algoritmos para redes com pesos arbitrários. O outro tipo de métodos pertence a grafos com pesos não negativos, e foram desenvolvidos algoritmos deste método por: Leyzoreck, Gray, Johnson, Ladew, Meaker, Petry e Seitz (1957) e por Dijkstra (1959) quem desenvolveu o primeiro algoritmo mais eficiente (Dreyfus 1969; Ahuja et al. 1990).

Em 1955 – 1957, o problema do caminho mais curto transformou-se num problema de programação linear. Por citar alguns autores, Orden (1955), Dantzing (1957) quem descreveu um procedimento gráfico do simplex aplicado a este problema. Ford (1956), Beckmann, McGuire e Winsten (1956), Bock e Cameron (1958), Bellman (1958), Minty (1958), Leyzoreck, Gay, Johnson, Ladew, Meakes, Petry e Seitz (1957).

Em tempos mais atuais destacam-se autores como Yen (1971), Martins (1984), Gallo e Pallotino (1986), Sheknar e Fetterer (1996), Goldfarb e Jin (1999).

Na atualidade existe uma grande quantidade de algoritmos de busca do caminho mais curto como fruto do trabalho de inúmeros pesquisadores que perseguiram o desenvolvimento de algoritmos cada vez mais eficientes e que basicamente consistem no aperfeiçoamento dos algoritmos tradicionais.

Para Dreyfus (1969), por causa da intensa busca de bons algoritmos para resolver o problema do caminho mais curto alguns autores podem ter sido omitidos ou alguns algoritmos ficaram sem ser conhecidos. Este mesmo autor faz uma advertência devido à grande quantidade de algoritmos existentes na literatura, nem sempre a eficiência deles foi comprovada ou eles têm erros, daí a importância de fazer uma busca séria e detalhada na hora de escolher algoritmos do caminho mais curto.

"Escolher um algoritmo adequado entre os numerosos algoritmos existentes na literatura, é uma etapa crítica [...]", (Zhan, 1997, p.1).

"Nos últimos anos tem se dedicado uma boa quantidade de esforço para o estudo do problema do caminho mais curto. Mais de 200 publicações, mas sabe-se pouco sobre sua eficiência relativa", (Pape, 1974, p.1).

---

[2] N.L. Biggs, E.K. Lloyd, R.J. Wilson, Graph Theory 1736 - 1936, Clarendon Press, Oxford, 1976.

[3] L. Rosenfeld, Unusual problems and their solutions by digital computer techniques, in: Proceedings of the Western Joint Computer Conference (San Francisco, California, 1956), The American Institute of Electrical Engineers, New York, 1956, pp. 79{82.

Glover et al. (1985), Hung e Divoky (1988), Ahuja et al. (1990), Cherkassky et al. (1993), Zhan e Noon (1998) realizaram avaliações do desempenho de algoritmos do caminho mais curto.

Neste artigo apresenta-se o primeiro resultado de um estudo maior que está sendo realizado com o intuito de avaliar alguns algoritmos do caminho mias curto.

No presente trabalho trata-se a parte teórica do problema do caminho mais curto com ênfase no algoritmo de Dijkstra e realiza-se sua implementação numérica para resolver duas aplicações com redes de pequeno porte. O trabalho está organizado como segue: na seção 2 apresentam-se a formulação do problema, a seção 3 contém os algoritmos de problema do caminho mais curto do método de rotulação com sua respectiva classificação. A seção 4 contém o algoritmo de Dijkstra e duas aplicações resolvidas com a implementação deste algoritmo. As seções 5 e 6 trazem, respectivamente, as conclusões do trabalho e as referências bibliográficas utilizadas neste artigo.

2. FORMULAÇÃO MATEMÁTICA DO PROBLEMA

Seja uma rede orientada G = (N, A) que contém um conjunto de nós N com |N| = n e um conjunto de arcos A com |A| = m. Cada arco está representado por um par ordenado de nós (i, j) e tem um peso c_{ij} (distância, custos, etc.) associado a cada arco $(i, j) \in A$.

Num arco (i,j) $\in$ A, i é o predecessor de j e denomina-se como p(j). Dois arcos são adjacentes se existe entre eles pelo menos um nó comum. A lista de adjacência de nós A(i) é o conjunto de nós adjacentes ao nó i, tal que $A(i) = \{(j \in N : (i, j) \in A \}$

O peso de um caminho orientando é a soma dos pesos de cada arco que pertence ao caminho.

Um grafo conectado que não contém nenhum ciclo define-se como árvore. Diz-se que um nó r é raiz da árvore se é possível construir um caminho desde r até todo nó da árvore. Seja T uma árvore com nó raiz r $\in N$, uma árvore de G. Para o nó raiz r, a árvore de menor caminho com raiz r é a árvore orientada tal que o único caminho na árvore desde o nó r até qualquer um outro nó é o caminho mais curto entre estes nós na rede original.

O problema do caminho mais curto pode-se formular matematicamente como um problema de programação linear, dado a seguir:

$$\text{Minimizar} \sum_{(i,j)\in A} c_{ij} x_{ij} \tag{2.1a}$$

sujeito a:

$$\sum_{\{j:(i,j)\in A\}} x_{ij} - \sum_{\{j:(j,i)\in A\}} x_{ji} = \begin{cases} n-1 & \text{para i = s} \\ -1 & \text{para todo i} \in \text{N - \{s\}} \end{cases} \tag{2.1b}$$

$$x_{ij} \geq 0, \text{ para todo (i,j)} \in A \tag{2.1c}$$

Para o estudo do problema do caminho mais curto assume-se que:

- Todos os pesos dos arcos são inteiros.
- A rede contém um caminho orientado desde o nó fonte até os outros nós da rede.
- A rede não contém um ciclo negativo.
- A rede é orientada.

3. ALGORITMOS DO MÉTODO DE ROTULAÇÃO

Nos trabalhos relacionados com a avaliação dos algoritmos do problema do caminho mais curto é freqüente encontrar a terminologia de algoritmos de rotulação permanente e de correção de rótulos. Esta terminologia foi introduzida por Gilsinn e Witzgall (1973) que foram os primeiros pesquisadores que realizaram estudos de comparação e avaliação de algoritmos de busca do caminho mais curto (Glover et al.1985).

Os algoritmos do método de rotulação (*labeling methods algorithms*) dividem-se em dois tipos: algoritmos de rotulação permanente (*labeling-setting algorithms*) e algoritmos de correção de rótulos (*labeling-correcting algorithms*).

O método de rotulação contém os algoritmos de caminho mais curto que começam com uma árvore T que depois é ampliada ou modificada em cada iteração, até que esta seja a árvore de caminho mínimo. Os dois tipos de algoritmos de rotulação utilizam este critério, ou seja, constroem uma árvore de caminho mínimo desde um nó raiz r até um nó i. O resultado final dos algoritmos está representado por um único caminho de r até i. Enquanto se faz a construção e melhora da árvore, três dados importantes são mantidos para cada nó i:

- A distância rotulada d(i)
- O nó predecessor de i p(i)
- O status do nó i $S(i) \in \{\text{não visto, visitado, examinado }\}$.

Define-se a distância rotulada do nó i, d(i), como a distância atual do nó raiz r até um nó i em alguma etapa da aplicação de um algoritmo de busca do caminho mais curto. Esta distância rotulada d(i) é usada para registrar o limite superior da distância do nó raiz r até um outro nó i na árvore. No final de todas as iterações a distância rotulada representa a menor distância de r até o no i.

O predecessor do nó i, p(i), armazena o nó que precede imediatamente ao nó i na árvore.

Um nó tem o status de não visto quando sua distância rotulada tem o valor de infinito (computacionalmente este valor está representado por uma quantidade muito grande). Um nó tem o status de visitado quando sua distância foi atualizada pelo menos uma vez, ou seja, a sua distância é diferente do infinito; e finalmente um nó é considerado examinado se este foi analisado com o fim de diminuir a distância rotulada para algum nó j tal que $(i, j) \in A$, e conseqüentemente a sua distância rotulada não pode ser mais atualizada.

O método de rotulação inicia com os seguintes dados:

$$d(i) = \infty$$
$$p(i) = \phi$$
$$S(i) = \text{não visto para cada nó i}$$
$$d(r) = 0$$
$$S(r) = \textit{visitado}$$

O método avança examinando aos nós visitados começando pelo nó raiz r até que o nó destino i é examinado e rotulado permanentemente (rotulação permanente) ou até que todos os nós sejam examinados e rotulados permanentemente (correção de rotulação). O método finaliza quando é determinada a árvore de caminho mínimo num subconjunto de nós (rotulação permanente) ou no conjunto inteiro de nós (correção de rotulação).

Se a distância rotulada ao nó j pode ser diminuída a árvore é modificada pela substituição dos predecessores do nó j pelo conjunto p(j) = i e o nó j, e é considerado rotulado.

Cada nó cuja distância rotulada foi melhorada é colocado numa lista de nós examinado e elegível *(scan eligible node list)*, representada por SE. Esta lista no inicio está vazia.

Em resumo examinar um nó i para os dois métodos de rotulação consiste em:

procedimento examinar (i)

para (i,j) $\in A$ fazer
se d(j) > d(i) + c_{ij} então
d(j) = d(i) + c_{ij}
S(j) = visitado
p(j) = i
S(i) = examinado
fim

A aplicação dos algoritmos de rotulação permanente pode ser feita a redes com arcos de pesos não negativos unicamente. Começam com uma árvore T que contém um nó raiz r e um conjunto de distâncias rotuladas tal que $d(r) = 0$ e $d(i) = \infty$.

O nó raiz e seus arcos associados são examinados com o objetivo de determinar se alguma distância pode ser melhorada. Esta etapa é chamada de examinado.

Cada nó cuja distância rotulada foi melhorada é colocado na lista SE, e seu arco associado é registrado. Em cada seguinte iteração o nó que pertence a SE com a mínima distância rotulada é selecionado, removido da lista e agregado à arvore T junto com seu arco associado. Este nó é examinado com o nó raiz, e depois a lista SE é atualizada. O algoritmo termina quando a lista SE fica vazia.

A propriedade fundamental dos algoritmos de rotulação permanente é que T é sempre a árvore de menor caminho desde r até todos os nós que pertencem a T em cada iteração. Por tanto o método de rotulação permanente desenvolve-se em exatamente |n| iterações e examina cada arco uma vez só. Pode-se parar o algoritmo depois de realizar as |n| iterações. É conveniente fazer isto especialmente quando se for trabalhar com o limite no pior caso. Observa-se que estes algoritmos só examinam os nós com distâncias rotuladas.

Uma característica dos algoritmos de rotulação permanente é que desenvolvem um procedimento que encontra o nó de mínima distância rotulada na lista de nós SE.

O primeiro algoritmo de rotulação permanente é atribuído a Dijkstra (1959) e Whiting e Hillier (1960) com uma complexidade no pior caso de $O(|n|^2)$.

As pesquisas realizadas no âmbito dos algoritmos de rotulação permanente concentram-se no subproblema de determinar o nó com a mínima distância rotulada que deve ser agregado à árvore T, (Glover et al., 1985).

Outros algoritmos de caminho mais curto do método de rotulação permanente foram desenvolvidos por Dial (1969), Johnson (1977), Denardo e Fox (1979).

Os algoritmos de correção de rótulos são mais gerais do que os algoritmos de rotulação permanente, pois podem ser aplicados a redes com arcos de pesos negativos além de que podem começar com qualquer árvore. Não obstante existem alguns algoritmos de correção de rótulos que são para ser usados especificamente com redes de pesos não negativos.

Em cada iteração, os algoritmos de correção de rótulos tentam melhorar a árvore T através da redução da distância rotulada de pelo menos um nó que lhe pertença com a agregação de um novo arco e um novo nó a T, esta agregação é feita substituindo um arco existente. O objetivo de fazer esta substituição é diminuir o caminho até o nó na árvore.

Com esta descrição pode-se já apreciar uma diferença entre os dois tipos de métodos de algoritmos rotulados. O critério de seleção de nós dos algoritmos de rotulação permanente é escolher o nó com a menor distância rotulada da lista de nós SE, enquanto os algoritmos de correção de rotulação podem selecionar qualquer nó desta lista.

O primeiro algoritmo de correção rotulada é atribuído a Bellman (1958), Ford (1956), e Moore (1959) e tem uma complexidade no pior dos casos de O(|n||a|), (Glover et al.1985). Neste algoritmo os nós são selecionados de lista de nós SE com o critério da regra primeiro em entrar primeiro em sair (FIFO, first-in-first-out). Yen (1970) fez modificações a este algoritmo.

Gilsinn e Witzgall (1973) desenvolveram um algoritmo de correção de rótulos utilizando o critério último em entrar primeiro em sair (LIFO *last-in-first-out*), neste

algoritmo os primeiros nós em ser agregados à árvore são aqueles que têm uma posição na frente da lista de nós SE e de igual forma estes são os primeiros em ser removidos.

D'Esopo e Pape (1974) desenvolveram um algoritmo utilizando um critério bidirecional que consiste em agregar um nó que ocupe um lugar na frente da lista de nós SE só se este aparecesse na lista antes de chegar ao fim, caso contrário os nós da frente da lista são removidos.

Glover, Klingman e Phillips (1985) desenvolveram um algoritmo denominado PSP (Partitioning Shortest Path)

4. ALGORITMO DE DIJKSTRA E APLICAÇÃO NUMÉRICA

4.1 ALGORITMO DE DIJKSTRA

O algoritmo de Dijkstra encontra o menor caminho de um nó fonte ou origem s até os outros nós de uma rede orientada para o caso em que todos os pesos dos arcos sejam não negativos. Mantém a distância rotulada d(i) para cada nó i, e é o limite superior do caminho mais curto até o nó i. O algoritmo divide os nós em dois grupos: rotulados permanentemente (ou permanentes) e rotulados temporariamente (temporários). A distância rotulada dos nós permanentes representa a menor distância do nó fonte até um outro nó. No caso dos nós temporários, esta distância representa o limite superior da distância do caminho mais curto até o nó.

O algoritmo examina todos os nós sem incluir o nó fonte s e rotula permanentemente os nós em ordem das distâncias ao nó fonte.

Inicia-se o algoritmo com d(s) = 0 e $d(i) = \infty$, para todo j que pertence à rede.

Em cada iteração o rótulo do nó i é o menor caminho desde o nó fonte ao longo de um caminho que contém outros nós intermediários. O algoritmo escolhe o nó i com a mínima distância rotulada temporária para fazê-la permanente e visita os outros nós, ou seja, examina os arcos de A(i) e atualiza as distâncias rotuladas dos nós adjacentes. O algoritmo termina quando todos os nós são designados como permanentes. A correção do algoritmo baseia-se na observação que é sempre possível designar o nó com a mínima distância temporária como permanente.

O algoritmo de Dijkstra finaliza quando não existirem nós com rótulos temporários (caminho mais curto do nó s para todos os outros) ou quando o rótulo do nó **i** passar a permanente (caminho mais curto do nó s para o nó i).

Este algoritmo mantém a árvore orientada T com raiz o nó fonte e que abrange os nós com distâncias finitas rotuladas.

Para $(i, j) \in T$, neste algoritmo pred(j) = i, é registrado, e para cada arco $(i, j) \in T$, $d(j) = d(i) + c_{ij}$. No fim do algoritmo quando as distâncias rotuladas representam os caminhos mais curtos, T é a árvore de caminho mínimo.

No algoritmo de Dijkstra a operação de seleção da mínima distância rotulada temporária conhece-se como "operação de seleção de nós". A operação de verificação de rótulos atuais dos nós i e j que satisfaçam a condição $d(j) > d(i) + c_{ij}$, e se isto acontece então $d(j) = d(i) + c_{ij}$, conhece-se como operação de "atualização de distâncias".

Pela forma como o algoritmo determina o caminho mais curto, o número de operações realizadas vem determinado a partir de duas operações: seleção de nós e atualização de distâncias.

Na seleção de nós esta operação é realizada em n nós e a partir de cada um destes são analisados (n-1) nós associados aos (n-1) arcos que saem de cada nó. Esta operação não é feita nos nós que têm o rótulo permanente por tanto a partir do k-ésimo nó selecionado são analisados n-k nós. Assim o número de operações da seleção de nós é:

$$n + (n-1) + (n-2) + \mathrm{K} + 1 = O(n^2)$$

A atualização de distâncias é realizada para cada nó durante |A(i)| vezes, e $\sum_{i \in N} |A(i)| = m$, então o algoritmo requer de executar esta operação m vezes, por tanto a complexidade é de O(m).Com a análise das duas componentes então a complexidade do algoritmo é:

$O(n^2) + O(m) = O(n^2)$, pois $m < n^2$.

A Figura 1 contém o algoritmo de Dijkstra:

começar
$S := \phi$
$\overline{S} := N$
$d(i) := \infty$ para cada nó $i \in N$
$d(s) := 0$
$pred(s) := 0$
se $|S| < n$ fazer
começar
Seja $i \in \overline{S}$ um nó tal que $d(i) = \min\{d(j) : j \in \overline{S}\}$;
$S := S \cup \{i\}$
$\overline{S} := \overline{S} - \{i\}$
para cada $(i, j) \in A(i)$ fazer
se $d(j) > d(i) + c_{ij}$ então $d(j) := d(i) + c_{ij}$ e $pred(j) := i$
fim
fim

Figura 1. Algoritmo de Dijkstra
Fonte: Ahuja et al. 1993, p. 109

4.2 APLICAÇÃO NUMÉRICA

Para explicar o procedimento do algoritmo de Dijkstra se utilizará a rede da Figura 2. Na Tabela 1 apresenta-se a cada iteração do algoritmo, os nós que são visitados, rotulados e que saem da lista SE.

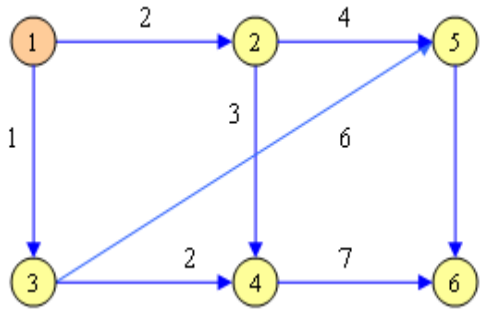

Figura 2 Rede orientada

Iteração No.	Nó candidato a ser visitado	Nós rotulados	Nó fora da lista SE
1	{ 1 }	(0, ∞,∞,∞,∞,∞)	1
2	{ 2, 3 }	(0,2,1, ∞,∞,∞)	3
3	{ 2, 4 }	(0,2,1,3, ∞,∞)	2
4	{ 4, 5 }	(0,2,1,3,6, ∞)	4
5	{ 5, 6 }	(0,2,1,3,6,10)	5
6	{ 6 }	(0,2,1,3,6,10)	6
	Φ	(0,2,1,3,6,10)	

Tabela 1. Iterações do Algoritmo de Dijkstra

Para a resolução de exemplos maiores realizou-se a implementação do algoritmo de Dijkstra num computador com processador Mobile AMD Sempron (tm) 795 MHz, 448 MB de RAM e trabalhou-se com o compilador e editor de FORTRAN, FORCE 2.0. Os resultados obtidos no programa realizado para a rede da Figura 2 são os seguintes:

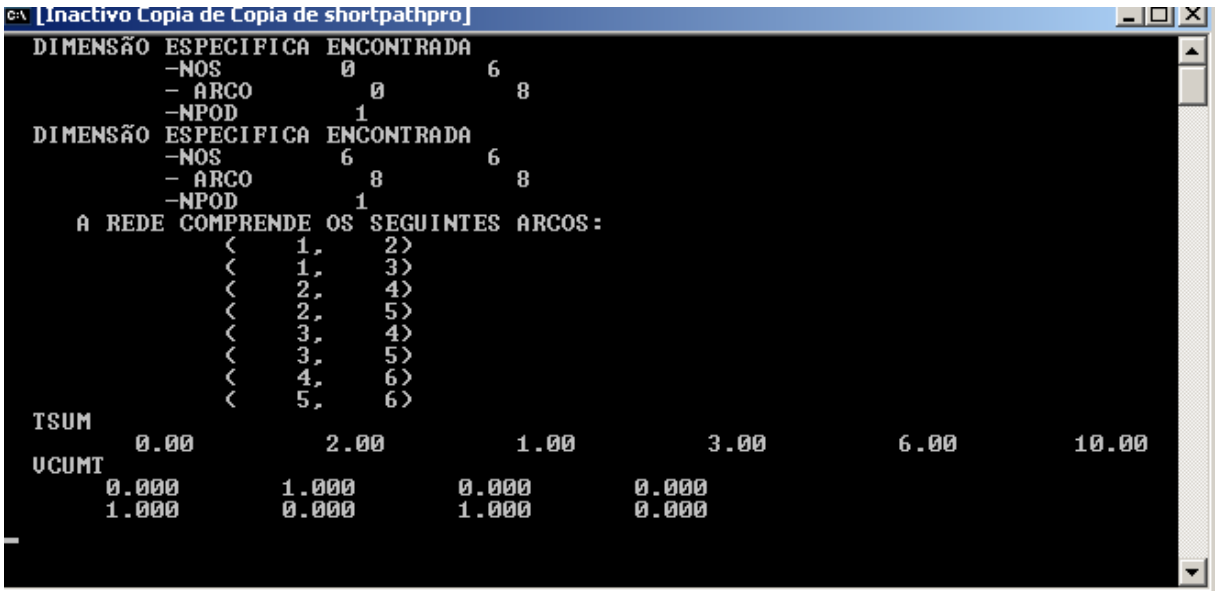

Figura 3 Resultados de implementação do algoritmo de Dijkstra

Dimensão específica contém o número de nós e arcos, 6 e 8 respectivamente. Depois aparecem cada um dos arcos da rede, arco 1 comprendido entre os nós (1, 2), arco 2 comprendido entre os nós (1, 3) e assim por diante até o arco 8 comprendido entre os nós (5, 6). TSUM contém as distâncias rotuladas mínimas obtidas a cada iteração (comparar com a tabela 1). Finalmente VCUMT indica os arcos que pertencem ao caminho mais curto, tal que

se o valor que aparece é 1, então o arco pertence ao caminho mais curto e 0 no caso contrário. Assim para a rede dada a solução contém os arcos 2, 5 e 7 (linha vermelha) assim:

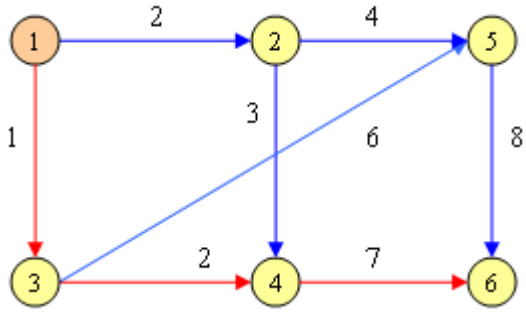

Figura 4 Caminho mínimo da rede 1 determinado a partir do algoritmo de Dijkstra

Também foi determinado o caminho mais curto numa rede que contém 20 nós e 30 arcos, os resultados obtidos com a implementação do algoritmo são os seguintes:

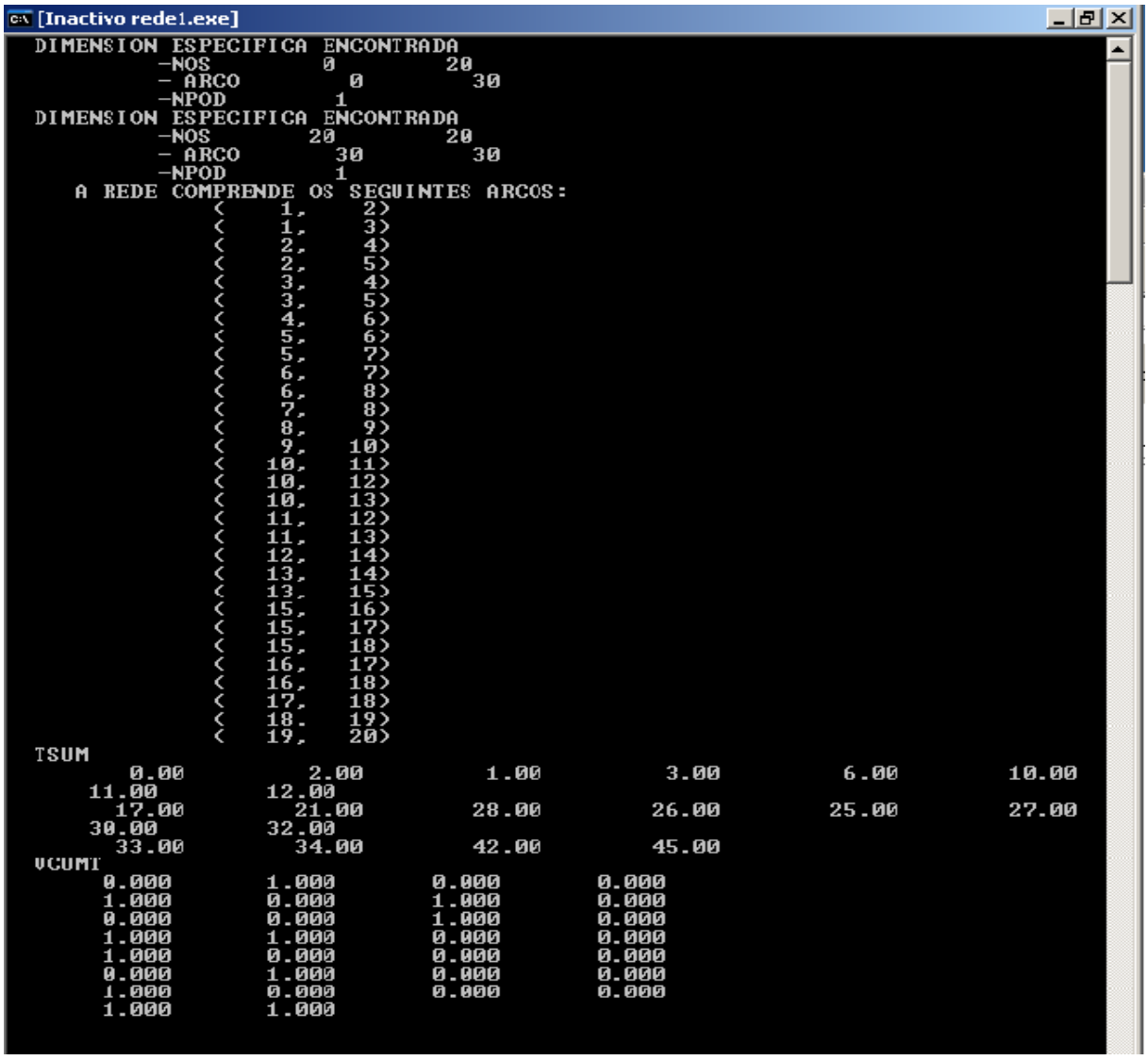

Figura 5 Caminho mínimo numa rede com 20 nós e 30 arcos, determinado a partir da implementação do algoritmo de Dijkstra

5. CONCLUSÕES

Neste artigo apresentou-se o problema do caminho mais curto. Os algoritmos para a solução deste problema classificam-se em dois tipos: algoritmos de rotulação permanente e de correção de rotulação. Os primeiros designam um nó rotulado permanente (ótimo) a cada iteração e são aplicáveis a redes com pesos de arcos não negativos. Os algoritmos do método de rotulação consideram todos os rótulos dos nós como temporários e somente ao finalizar o algoritmo estes são designados como permanentes, estes algoritmos são aplicáveis a redes com pesos de arcos arbitrários.

Neste trabalho foi descrito o algoritmo de Dijkstra, considerado como um dos algoritmos mais eficientes. Este algoritmo é do método de correção permanente e é aplicável em redes que contenham pesos de arcos não negativos e sua complexidade é de $O(n^2)$.

No entanto é importante mencionar que devido às diversas aplicações práticas do problema do caminho mais curto existem alguns algoritmos para resolvê-lo e por tanto é muito grande a variedade de algoritmos que podem ser usados.

Atualmente estamos elaborando uma pesquisa que compreende a avaliação de alguns algoritmos do caminho mais curto que serão testados com redes aleatórias de grande porte.

6. REFERÊNCIAS BIBLIOGRÁFICAS

[1] AHUJA, R., MAGNANTI, T., ORLIN, J., Network Flows Theory: algorithms and applications, 1993

[2] AHUJA, R., MEHLHORN, K., ORLIN, J., TARJAN, R., Faster Algorithms for the shortest path, 1990. Disponível em: <http://www.cs.umd.edu/~gasarch/651/dijcjacm.pdf>. Aceso em: 11 nov. 2007.

[3] CHERKASSKY, B. V., GOLDBERG A. V., RADZIK, T., (1996) Shortest Paths Algorithms:
Theory and Experimental Evaluation, Mathematical Programming, 73, 129-174.

[4] DIJKSTRA, W., A note on two problems in connection with graphs. Num. Math., 1 (1959), pp. 269 - 271

[5] HUNG, M., DIVOKY,J., Performance of shortest path algorithms in network flow problems, Management Science, 1990.

[6] DREYFUS, S, An appraisal of some shortest path algorithms. Operations Research, 17 (1969), pp. 395 -411

[7] EPPSTEIN, D., Finding the k Shortest Paths. Tech. Report 94-26, Univ. of California, Irvine, Dept. of Information and Computer Science, (1994), p. 23. Disponível em : <http://www.ics.uci.edu/~eppstein/pubs/Epp-TR-94-26.pdf>. Aceso em: 18 nov. 2007.

[8] GLOVER, F., KLINGMAN, D., PHILLIPS N., SCHNEIDER, R., New Polynomial Shortest path algorithms and their computational attributes, Management Science (pre-1986); Sep 1985; Vol 31, No 9; ABI/INFORM Global p. 1106 – 1128.

[9] HILLIER, F. S., LIEBERMAN G. J.. Introdução à pesquisa operacional. 3. Ed. Rio de Janeiro: Campus / São Paulo: Universidade de São Paulo, 1988. 805 p.

[10] PAPE, U., IMPLEMENTATION AND EFFICIENCY OF MOORE-ALGORITHMS FOR THE SHORTEST ROUTE PROBLEM, Mathematieal Programming 7 (1974) 212-222. North-Holland Publishing CompanyTechnische Universitd't Berlin, W. Berlin, Germany

Disponível em: <http://www.springerlink.com/content/u6731511r0446472/>. Aceso em 02 mar. 2008.

[11] SCHRIJVER, A., On the history of combinatorial optimization (till 1960), ``Handbook of Discrete Optimization'' (K. Aardal, G.L. Nemhauser, R. Weismantel, eds.), Elsevier, Amsterdam, 2005, pp. 1- 68.

[12] ZHAN, B., Three Fastest Shortest Path Algorithms on Real Road Networks: Data Structures and Procedures : Journal of Geographic Information and Decision Analysis, vol.1, no.1, (1997) pp. 69-82. Disponível em:
< http://publish.uwo.ca/~jmalczew/gida_1/Zhan/Zhan.htm>. Aceso em 21 dez. 2007.

[13] ZHAN, F. B., NOON, C. E. (1998) Shortest Path Algorithms: An Evaluation Using Real Road Networks, Transportation Science, 32, 65-73.