Uma Implementação de Pontos Interiores para Fluxo em Redes Lineares por Partes

Clovis Perin
IMECC–Unicamp

Margarida Mello
IMECC–Unicamp

Fernando A.S. Marins
FEG–Unesp

Novembro, 1999

Resumo

Problemas de Fluxo em Redes Lineares por Partes constituem uma importante área de aplicação, onde são estudados modelos que tratam da minimização de uma função objetivo convexa, separável e linear por partes, sujeita a um sistema de equações lineares que expressam a conservação de fluxo sobre uma rede conectada. Este problema pode ser transformado em um programa linear equivalente de tamanho maior. Neste trabalho apresentamos e comparamos duas implementações do método primal-dual para redes: uma para problemas lineares e outra para problemas lineares por partes. Ambas as implementações utilizam estruturas de dados especiais visando diminuir o gasto com memória e tempo. A estrutura utilizada para redes lineares por partes pode ser considerada uma extensão daquela utilizada em redes lineares. Cabe observar que o esforço computacional de cada iteração está concentrado na construção e resolução de um sistema linear quadrado, simétrico e definido positivo, resolvido por um método de gradientes conjugados precondicionado. Além disto, a matriz do sistema não é construída explicitamente. Foram realizados diversos testes computacionais que permitem concluir afirmativamente a respeito da vantagem computacional da segunda implementação.

1 Introdução

Considera-se um problema de minimização com função objetivo convexa, separável e linear por partes, e com restrições que representam a conservação de fluxo em uma rede conectada. Este modelo pode surgir em um contexto teórico, quando se estudam métodos para obtenção de solução inicial para problemas lineares, ou diretamente de aplicações práticas, por exemplo nas áreas de telecomunicações, planejamento de operação de usinas hidrelétricas e distribuição de energia elétrica. No primeiro caso a rede matemática é um retrato direto da rede telefônica e a quantidade de fluxo que trafega por um arco corresponde à intensidade de tráfego telefônico entre os nós ligados por este arco. Como neste tipo de problema geralmente admite-se a possibilidade de expansão de capacidade de determinados arcos, é natural modelar esta situação atribuindo-se uma estrutura de custos linear por partes para os arcos. Assim, os custos são lineares para determinadas faixas de fluxo com coeficientes maiores para as faixas mais elevadas. A importância e a ampla utilização destes modelos decorrem das propriedades de facilidade de visualização da solução, flexibilidade dos modelos e disponibilidade de métodos eficientes para resolvê-los. Para tanto têm sido desenvolvidas estruturas de dados específicas e algoritmos de programação linear têm sido adaptados. Em particular, o método simplex para redes, que já é uma adaptação eficiente do método simplex para programas lineares, foi adaptado com sucesso para redes lineares por partes, ver [3, 6]. Após o advento dos algoritmos de pontos interiores para programação linear, e.g. [9], surgiram iniciativas no sentido de formular uma variante eficiente para problemas em redes, cf. [11]. Neste trabalho desenvolvemos uma adaptação bem sucedida de pontos interiores para redes lineares por partes, considerando uma estrutura especial de dados, prosseguindo com trabalho já realizado em [7, 8, 10].

O custo linear por partes de uma variável é expresso por meio de duas seqüências crescentes de números: a primeira contendo os extremos dos intervalos consecutivos dentro dos quais o custo é linear e a segunda com os coeficientes de custo para cada intervalo. Ho, em [4], apresenta os quatro métodos mais usuais para transformar um problema linear por partes em um problema linear equivalente, trabalhando cada variável cujo custo é linear por partes da seguinte forma:

- no δ-Método (*sum-of-intervals* ou *bounded variables representation*) a variável é substituída pela soma de outras variáveis, uma para cada intervalo e o custo associado à variável original é substituído por uma combinação linear das

novas variáveis cujos multiplicadores são os coeficientes de custo dos intervalos correspondentes;

- no λ-Método (*weighted-grid-point* ou *inner approximation*) a variável é substituída pela combinação convexa dos extremos dos intervalos e o custo associado à variável original é substituído pela mesma combinação convexa dos custos correspontes aos extremos dos intervalos;
- o σ-Método (*outer approximation*) utiliza o fato que uma função convexa e linear por partes pode ser expressa como o máximo pontual de funções lineares
- no γ-Método (*sum-of-projections representation*) o custo é calculado com o efeito cascata, conhecido por todos os declarantes de IR.

Os autores [10] compararam o uso dessas transformações em problemas em redes lineares por partes. O método mais promissor, segundo este estudo, foi o δ-Método. Ademais, este é o único método que preserva a estrutura de rede da matriz de coeficientes.

Conseqüentemente, este foi o método adotado para comparar as duas abordagens de resolução via pontos interiores: (i) transformando-se o problema linear por partes em um problema linear equivalente e aplicando-se a este último um método de pontos interiores para redes lineares e (ii) resolvendo-se o problema linear por partes por uma variante de método de pontos interiores capaz de lidar diretamente, sem a necessidade de transformações, com redes lineares por partes.

Este trabalho insere-se, portanto, em uma linha de pesquisa mais ampla, na qual são exploradas várias facetas das possíveis abordagens do problema de redes lineares por partes: os tipos de algoritmos que podem ser utilizados, as transformações, as estruturas de dados, os detalhes de implementação relevantes (e.g., rotinas para a resolução de sistemas lineares), etc., com o objetivo de se determinar qual a abordagem ideal para este tipo de problema.

O trabalho apresenta-se organizado como segue: na seção 2 o arcabouço dos algoritmos de pontos interiores implementados é descrito, na seção 3 discutem-se os detalhes das implementações descrevendo-se as estruturas de dados utilizadas. Os testes computacionais são relatados na seção 4 e as conclusões e direções para trabalhos futuros são resumidas na seção 5

2 Desenvolvimento e Descrição do Algoritmo

Neste trabalho mostramos que é interessante considerar a implementação especializada de um método de pontos interiores para problemas de fluxo em redes lineareas por partes ao invés de considerar a sua transformação em um problema de fluxo em redes lineares.

Diversos métodos de primais-duais de pontos interiores já foram propostos e os pontos centrais de discussão podem ser resumidos em: (1) A cada iteração primal-dual, resolver um sistema simétrico e definido positivo de equações lineares ou resolver o sistema aumentado equivalente? (2) No caso de resolver o sistema simétrico e definido positivo, utilizar a fatoração de Cholesky, Gradientes Conjugados, ou algum outro método? (3) No caso de precondicionamento para o método dos gradientes conjugados utilizar o precondicionador diagonal, da árvore geradora máxima, ou algum outro?

Diversos autores, e.g. [11], mostraram que é eficiente resolver o problema de fluxo em redes lineares computando, a cada iteração do método primal-dual de pontos interiores, a solução de um sistema simétrico e definido positivo com o método dos gradientes conjugados precondicionado pelo precondicionador diagonal, nas iterações iniciais, e pelo precondicionador da árvore geradora máxima, nas iterações subseqüentes do método primal-dual. Vamos utilizar esta mesma metodologia para resolver problemas de fluxo em redes lineares por partes.

O problema de fluxo em redes lineares por partes definido por

$$\min\{f(x) \mid Ax = b, x \geq 0\},$$

onde A é a $n \times m$-matriz de incidência de uma rede com n nós e m arcos, b é o n-vetor de demandas dos nós, x é um m-vetor de fluxos nos arcos, a ser determinado, e $f(x) = \sum_{j=1}^{m} f_j(x_j)$ é uma função convexa, separável e linear por partes. As funções convexas e lineares por partes f_j podem ser especificadas pelo par de ℓ-vetores c, d, onde $\ell = \sum_{j=1}^{m} \ell_j$ é o número total de intervalos, sendo ℓ_j o número de intervalos de f_j.

Este problema pode ser transformado em um problema de fluxo em redes lineares definido por

$$\min\{c'\bar{x} \mid \bar{A}\bar{x} = b, \bar{x} + \bar{s} = d, \bar{x}, \bar{s} \geq 0\}$$

Neste caso, a $n \times \ell$-matriz $\bar{A}$ é obtida a partir da matriz A com a replicação de

cada uma de suas colunas tantas vezes quantos são os intervalos a ela associados. Estamos supondo que esta replicação também é feita no vetor x transformando-o no vetor $\bar{x}$. O problema dual associado é dado por

$$\max\{b'y - d'\bar{w} \mid \bar{A}'y - \bar{z} + \bar{w} = c, \bar{z}, \bar{w} \geq 0\}$$

e as condições de folgas complementares são expressas por

$$\overline{X}\,\overline{Z}\mathbf{1} = 0 \quad \overline{S}\,\overline{W}\mathbf{1} = 0$$

onde $\overline{X}, \overline{S}, \overline{Z}, \overline{W}$ são matrizes diagonais com os elementos de $\bar{x}, \bar{s}, \bar{z}, \bar{w}$, respectivamente, e $\mathbf{1}$ é um vetor de 1's.

O método primal-dual de pontos interiores adotado é inicializado a partir de um iterando $\bar{x}, \bar{s}, y, \bar{z}, \bar{w}$ satisfazendo $\bar{x}, \bar{s}, \bar{z}, \bar{w} > 0$ e a cada iteração é obtida uma direção de deslocamento $\Delta\bar{x}, \Delta\bar{s}, \Delta y, \Delta\bar{z}, \Delta\bar{w}$ em conformidade com um parâmetro de centragem μ de acordo com as equações abaixo que refletem a factibilidade primal, a factibilidade dual, e a complementaridade das soluções quando $\mu = 0$.

$$\begin{aligned}
\bar{A}\Delta x &= b - \bar{A}\bar{x} \\
\Delta\bar{x} + \Delta\bar{s} &= d - \bar{x} - \bar{s} \\
\bar{A}'\Delta y + \Delta\bar{z} - \Delta\bar{w} &= c - \bar{A}'y - \bar{z} + \bar{w} \\
\overline{X}\Delta\bar{z} + \overline{Z}\Delta\bar{x} &= \mu\mathbf{1} - \overline{X}\bar{z} \\
\overline{S}\Delta\bar{w} + \overline{W}\Delta\bar{s} &= \mu\mathbf{1} - \overline{S}\bar{w}
\end{aligned}$$

Isto resulta na seguinte seqüência de comandos:

$$\begin{aligned}
D &\leftarrow (\overline{X}^{-1}\overline{Z} + \overline{S}^{-1}\overline{W})^{-1} \\
Q &\leftarrow \bar{A}D\bar{A}' \\
q &\leftarrow b - \bar{A}\bar{x} + \bar{A}D(c - \bar{A}'y - \mu(\overline{X}^{-1} - \overline{S}^{-1})\mathbf{1} - \overline{S}^{-1}\overline{W}(d - \bar{x} - \bar{s})) \\
\Delta y &\leftarrow \text{solução do sistema } Q\Delta y = q \\
\Delta\bar{x} &\leftarrow D(\bar{A}'\Delta y - c + \bar{A}'y + \mu(\overline{X}^{-1} - \overline{S}^{-1})\mathbf{1} + \overline{S}^{-1}\overline{W}(d - \bar{x} - \bar{s})) \\
\Delta\bar{s} &\leftarrow d - \bar{x} - \bar{s} - \Delta\bar{x} \\
\Delta\bar{z} &\leftarrow \mu\overline{X}^{-1}\mathbf{1} - \bar{z} - \overline{X}^{-1}\overline{Z}\Delta\bar{x} \\
\Delta\bar{w} &\leftarrow \mu\overline{S}^{-1}\mathbf{1} - \bar{w} - \overline{S}^{-1}\overline{W}\Delta\bar{s}
\end{aligned}$$

O algoritmo pára quando uma das seguintes condições é satisfeita: (1) número máximo de iterações é excedido; (2) soluções quase-complementares $\bar{x}'\bar{z} + \bar{s}'\bar{w} \approx 0$

são encontradas; (3) limite inferior para o valor da função objetivo primal é excedido; (4) limite superior para o valor da função objetivo dual é excedido. O parâmetro μ é atualizado dividindo-se a complementaridade média corrente pelo número de variáveis.

O método primal-dual de pontos interiores pode ser esquematizado da seguinte forma:

```
let  x̄, s̄, y, z̄, w̄  such that x̄, s̄, z̄, w̄ > 0
while  (  critério de parada não é satisfeito  )  do
   update μ
   obtain Δx̄, Δs̄, Δy, Δz̄, Δw̄
   find α  such that (x̄, s̄, z̄, w̄) + α(Δx̄, Δs̄, Δz̄, Δw̄) > 0
   update (x̄, s̄, y, z̄, w̄) ← (x̄, s̄, y, z̄, w̄) + α(Δx̄, Δs̄, Δy, Δz̄, Δw̄)
```

Destacamos a seguir os aspectos que são diferentes entre as duas versões que são implementadas do algoritmo, para redes lineares e para redes lineares por partes. Na especialização do algoritmo para redes lineares por partes trabalha-se com um vetor adicional, um vetor de escala para os arcos, além do vetor de escala para os intervalos, com vistas a reduzir o esforço computacional. Na comparação abaixo veremos como o uso deste vetor permite reduzir o número de operações, enfocando, em particular, o esforço dispendido na execução do gradiente conjugado, que é efetuada a cada iteração:

(1) O produto $u \leftarrow Qv$ é computado em cada iteração do método do gradiente conjugado precondicionado para a resolução do sistema $Q\Delta y = q$. A matriz simétrica e definida positiva $Q = \bar{A}D\bar{A}'$ é manipulada implicitamente da seguinte forma: o vetor u, que é inicializado com zeros, é construído somando-se parcelas apropriadas, uma por arco (para redes lineares por partes) ou por intervalo (para redes lineares). No caso de redes lineares por partes, obtém-se a diferença entre as componentes de v da cabeça e da cauda de cada arco, multiplica-se esta diferença pelo fator de escala do arco e soma-se na componente de u da cabeça do arco e subtrai-se na componente de u da cauda do arco. No caso de redes lineares, faz-se a mesma operação, porém todas as réplicas dos arcos devem ser consideradas, portanto ao invés de lidarmos com m arcos, estaremos lidando com ℓ réplicas de arcos.

(2) O precondicionador da árvore geradora máxima nas aplicações do método do gradiente conjugado com este precondicionador é determindado da seguinte forma: constrói-se o monte (*head*) de ordenação dos arcos (ou intervalos) de acordo com o seu fator de escala e retira-se do monte os arcos (ou intervalos) de maior fator de escala até se obter uma árvore geradora. Esta árvore corresponde à base de uma solução extrema próxima da solução corrente e tem a propriedade de ser triangularizável. Note-se que a dimensão do monte para redes lineares é de ℓ elementos (intervalos ou réplicas de arcos) enquanto que, para redes lineares por partes, o monte tem dimensão m.

Cabe observar que nas duas implementações, para redes lineares e para redes lineares por partes, trabalhamos com um vetor solução ℓ dimensional ($\bar{x}$). Outros trabalhos, e.g. [1], adotam estratégia diferente, trabalhando com vetor solução m dimensional.

3 Implementação

Os programas foram implementados em C e os testes computacionais foram realizados em uma estação de trabalho SPARCStation 5. A estrutura de dados utilizada na implementação para pontos interiores utiliza vetores de tamanho definidos por n nós, m arcos e ℓ intervalos:

- armazenamento da rede linear: 1 n-vetor para armazenar b, 2 ℓ-vetores, para armazenar a cabeça e a cauda de cada réplica de arco e 2 ℓ-vetores para armazenar c, d.
- armazenamento da rede linear por partes: 1 n-vetor para armazenar b, 2 m-vetores para armazenar a cabeça e a cauda de cada arco e 3 ℓ-vetores para armazenar c, d e o arco de cada intervalo.
- armazenamento da solução e variáveis auxiliares: 2 n-vetores para armazenar os vetores $y, \Delta y, \hat{b}$, 10 ℓ-vetores para armazenar $\bar{x}, \bar{s}, \bar{z}, \bar{w}, \Delta\bar{x}, \Delta\bar{s}, \Delta\bar{z}, \Delta\bar{w}, \hat{c}$, d^{ℓ} e, no caso de redes lineares por partes, mais 1 m-vetor para armazenar d^{m}, o fator de escala de cada arco. Os vetores $\bar{s}, \bar{w}, \Delta\bar{s}, \Delta\bar{w}$ têm a sua dimensão reduzida para o número de intervalos com canalização finita.

- variáveis auxiliares para o gradiente conjugado: 4 n-vetores para armazenar a solução, o resíduo, a direcão de deslocamento da solução e a direção de redução do resíduo.
- precondicionamento diagonal: 1 n-vetor para armazenar d^n.
- precondicionamento da árvore geradora máxima: 4 n-vetores para armazenar o predecessor, fio (*thread*), fio reverso e sentinela de nó, e 1 m-vetor para armazenar o monte (*heap*) de ordenação.
- Obs.: $\hat{b} = b - Ax$, $\hat{c} = c - A'y - z + w$, d^ℓ é o fator de escala de cada intervalo, d^m é o fator de escala de cada arco otido com a soma dos fatores de escala de seus intervalos e d^n é o fator de escala de cada nó obtido com a soma dos fatores de escala dos arcos incidentes.

As tolerâncias utilizadas, em ambas as implementações, foram: 10^{-8} (para tolerância relativa no teste de complementaridade), -10^8 (para limite inferior da função objetivo primal) e 10^8 (para limite superior da função objetivo dual). O número máximo de iterações, em ambas as implementações, é de 100.

4 Testes Computacionais

Foram geradas aleatoriamente redes conectadas com 1000 a 10000 nós, 2000 a 50000 arcos e 4000 a 224000 intervalos. Os tempos de CPU (em segundos) e o número de iterações gasto por cada programa para cada rede estão relacionados nas Tabelas 1, 2, 3, 4 e 5, onde a terminação "L" está associada ao algoritmo para redes lineares e a terminação "P" ao algoritmo para redes lineares por partes. Estes números correspondem a médias, foram gerados 3 exemplares factíveis para cada tamanho de rede (nós, arcos e intervalos). É possível observar que, embora o número de iterações não apresente grandes diferenças, o tempo de CPU é sensivelmente menor para o algoritmo adaptado para redes lineares por partes. A média das taxas cpuL/cpuP para todos os testes é aproximadamente 1,5, ou seja, a estratégia de resolução que consiste em resolver o problema linear equivalente resulta em aproximadamente 50% a mais de tempo computacional para a resolução do problema.

Os dados da Tabela 1 referentes aos tempos de CPU estão organizados em forma de gráfico na figura 1. Esta representação gráfica pretende facilitar a percepção da

no. arcos	no. interv.	cpuL	cpuP	itrL	itrP
2000	4000	65	51	65	65
2000	10000	65	43	49	49
2000	16000	72	42	42	42
3500	7000	64	48	53	53
3500	17500	75	46	40	40
3500	28000	98	53	38	38
5000	10000	65	47	45	45
5000	25000	93	54	39	39
5000	40000	130	69	39	39

Tabela 1: Resultados experimentais para redes com 1000 nós

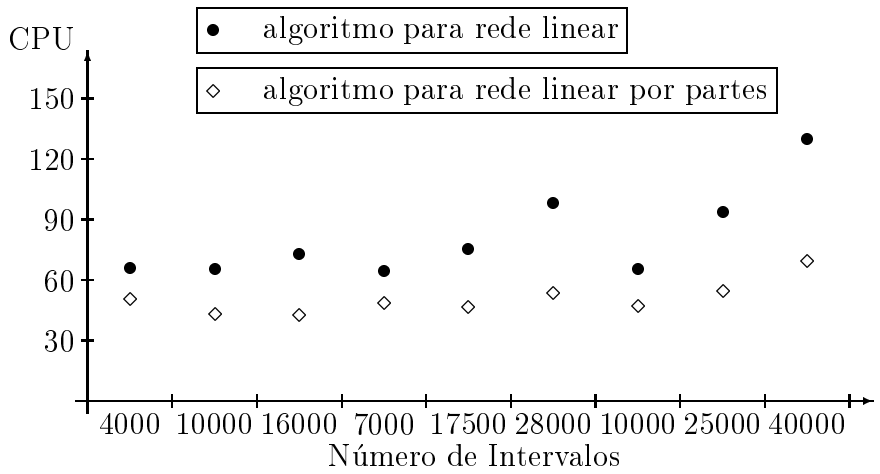

Figura 1: Tempos de cpu para redes com 1000 nós

diferença relativa entre os tempos.

no. arcos	no. interv.	cpuL	cpuP	itrL	itrP
4000	8000	240	195	79	79
4000	20000	215	148	56	56
4000	32000	219	137	48	48
7000	14000	224	175	63	63
7000	35000	216	138	44	44
7000	56000	275	157	43	43
10000	20000	226	167	54	54
10000	50000	260	158	42	42
10000	80000	373	196	43	43

Tabela 2: Resultados experimentais para redes com 2000 nós

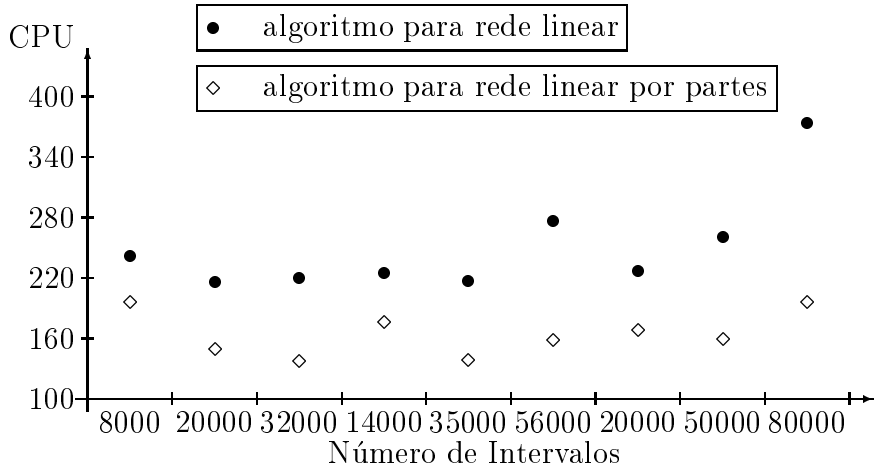

Figura 2: Tempos de cpu para redes com 2000 nós

Convém observar que, como era de se esperar, a superioridade da estratégia de resolver o problema por um algoritmo especializado para redes lineares por partes, fica mais evidente quanto maior o número de intervalos. Se comparamos os tempos de CPU para redes com mesmo número de arcos, porém com número crescente de intervalos (note que há três números de intervalos para cada número de arcos), vemos que a distância entre o círculo cheio "•" e o losango "◇" fica maior. Este fenômeno está presente em todos os gráficos.

no. arcos	no. interv.	cpuL	cpuP	itrL	itrP
8000	16000	948	788	92	92
8000	40000	774	570	67	67
8000	64000	748	494	55	55
14000	28000	862	697	76	76
14000	70000	699	478	50	50
14000	112000	780	468	44	44
20000	40000	795	618	63	63
20000	100000	821	504	46	46
20000	160000	1068	592	48	48

Tabela 3: Resultados experimentais para redes com 4000 nós

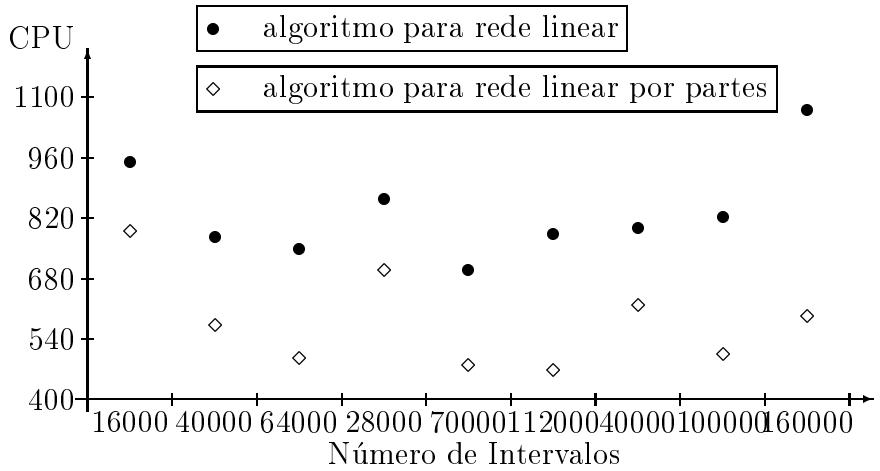

Figura 3: Tempos de cpu para redes com 4000 nós

no. arcos	no. interv.	cpuL	cpuP	itrL	itrP
16000	128000	2875	2051	67	67
28000	56000	3338	2837	90	90
28000	140000	2585	1856	57	57
28000	224000	2606	1679	50	50
40000	80000	2909	2351	71	71
40000	200000	2699	1807	51	51

Tabela 4: Resultados experimentais para redes com 8000 nós

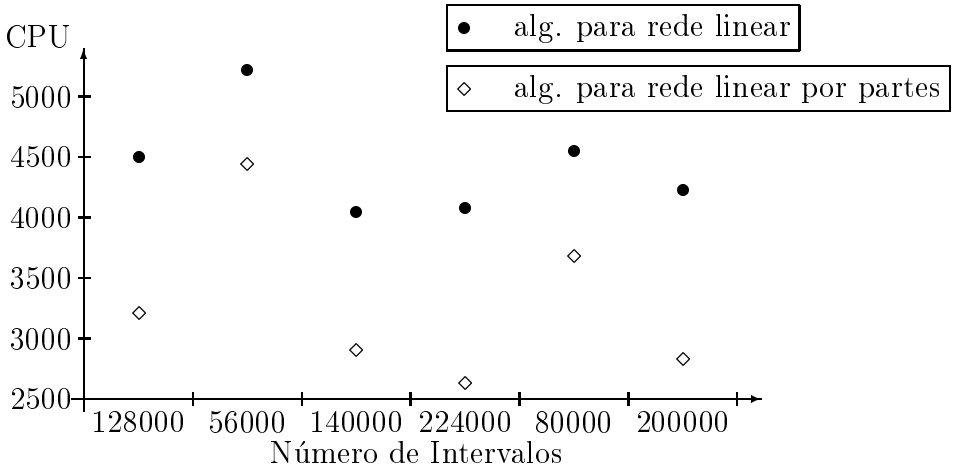

Figura 4: Tempos de cpu para redes com 8000 nós

no. arcos	no. interv.	cpuL	cpuP	itrL	itrP
20000	100000	4647	3602	80	80
20000	160000	4334	3201	69	69
35000	70000	4940	4236	88	88
35000	175000	3920	2933	61	61
50000	100000	4615	3768	77	77

Tabela 5: Resultados experimentais para redes com 10000 nós

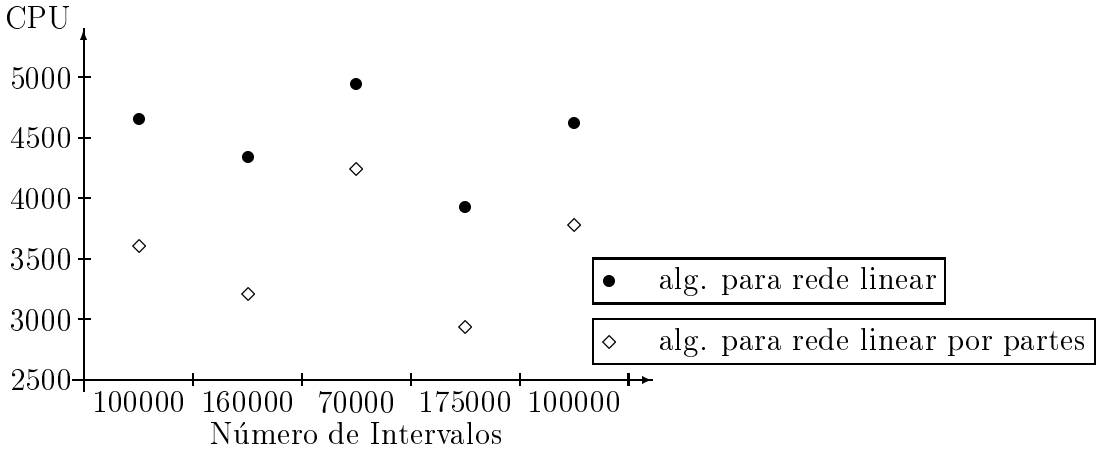

Figura 5: Tempos de cpu para redes com 10000 nós

5 Comentários Finais

Apresentamos e comparamos duas implementações de métodos de ponto interiores para redes, lineares e lineares por partes, respectivamente. Comparamos então duas estratégias para a resolução de problemas de fluxo lineares por partes:

(1) Converter o problema linear por partes em outro linear equivalente e aplicar o algoritmo para problemas lineares.

(2) Resolver o problema por um algoritmo especializado, capaz de atacar o problema linear por partes diretamente.

A investigação do método mais adequado de conversão a ser utilizado em (1) foi objeto de estudo anterior pelos autores [10]. No presente trabalho foi utilizado o δ-Método, que consiste em substituir cada variável cujo custo é linear por partes por uma soma de variáveis, uma para cada intervalo em que o custo é linear. Esta transformação apresenta, entre outras, a vantagem de preservar a estrutura de rede do problema, a matriz de coeficientes do problema linear equivalente continua sendo uma matriz de incidência de uma rede.

Elaboramos uma versão do método de pontos interiores para redes lineares adaptada para a resolução de redes lineares por partes. As implementações dos algoritmos foram testadas em 114 problemas gerados aleatoriamente. Ambos os algoritmos utilizam as mesmas regras de parada, ou seja, o controle de qualidade da solução que será aceita como ótima tem o mesmo rigor em ambos os casos. Os testes comprovam a superioridade da estratégia (2), sendo os tempos de CPU aproximadamente 50% maiores na estratégia (1).

Os autores investigam atualmente a adaptação do programa PCx, baseado em métodos de pontos interiores, para a resolução de problemas lineares para verificar se a superioridade da estratégia (2) permanece válida para problemas lineares gerais.

Referências

[1] C. Cavichia, Resolução de Problemas de Programação Linear por Partes via Algoritmos de Pontos Interiores, Tese de Doutorado, Faculdade de Engenharia

Elétrica e de Computação, Unicamp, 1997.

[2] J. Czyzyk, S. Mehrotra, M. Wagner e S.J. Wright, PCx user guide, Technical Report OTC 96/01, Optimization Technology Center, 1997.

[3] K. Darby-Dowman, F.A.S. Marins, E. Senne, C. Perin e A. Machado, Algorithms for network piecewise-linear programs: a comparative study, *European Journal of Operational Research* **97** (1997), 183–199.

[4] J.K. Ho, Relationships among linear formulations of separable convex piecewise linear programs, *Mathematical Programming Study* **24** (1985), 126–140.

[5] F.A.S. Marins e C. Perin, Programação linear por partes, *Produção* **6** (1996), 146–163.

[6] F.A.S. Marins e C. Perin, Algoritmos para programas em redes lineares por partes, *Pesquisa Operacional* **15** (1995) 67–90.

[7] F.A.S. Marins e C. Perin, An interior point approach to piecewise-linear programs, in "Anais de Resumos do Euro XV/Informs XXXIV Joint International Conference", Barcelona, Espanha, 1997.

[8] F.A.S. Marins e C. Perin, A primal-dual interior point method for network piecewise-linear programs, in "Anais de Resumos do 16th International Symposium on Mathematical Programming", Lausanne, Suíça, 1997.

[9] S. Mehotra e J. Wang, Conjugate gradient based implementation of an interior point method for network flow problems, Technical Report 95-70, Northwestern University, Evanston, IL, USA, 1995.

[10] C. Perin, F.A.S. Marins e M.P. Mello, Solução de programas lineares por partes em redes via pontos interiores, in "Anais de Resumos do XXI CNMAC", Caxambu, MG, 1998.

[11] M. Resende e G. Veiga, An efficient implementation of a network interior point method, Technical Report, AT&T Bell Laboratories, Murray Hill, NJ, USA, 1992.

[12] Y. Zhang, Solving large-scale linear programs by interior point methods under the MATLAB environment, Technical Report, Department of Compputational and Applied Mathematics, Rice University, Houston, TX, USA, 1997.