Fluxo de Custo Mínimo com Escala nos Custos

Autor: Mário César San Felice

Fonte: Livro de Ahuja, Magnanti e Orlin.

2 de dezembro de 2010, IC-Unicamp.

Estrutura da Apresentação

- Notações e Suposições
- Otimalidade Aproximada
- Descrição do Algoritmo
- Análise do Algoritmo

Notações e Suposições

- $G = (N, A)$ é nosso grafo dirigido.
- Para toda aresta $(i, j) \in A$ temos $c_{ij} \geq 0$ e $u_{ij} \geq 0$.
- Para todo vértice $i \in N$ temos oferta $b(i)$.
- Custos, capacidades e ofertas são inteiros.

Notações e Suposições

- $C = \max_{(i,j)\in A} c_{ij}$.
- $U = \max\{\max_{(i,j)\in A} u_{ij}, max_{i\in N}|b(i)|\}$.
- $\sum_{i\in N} b(x) = 0$.
- O problema tem solução viável.

Notações e Suposições

- $G(x) = (N, A(x))$ é o digrafo residual de G e x.
- Para toda aresta $(i, j) \in A$, se $x_{ij} < u_{ij}$ então (i, j) está em $A(x)$ com custo c_{ij} e capacidade $r_{ij} = u_{ij} - x_{ij}$.
- Para toda aresta $(i, j) \in A$, se $x_{ij} > 0$ então (j, i) está em $A(x)$ com custo $-c_{ij}$ e capacidade $r_{ji} = x_{ij}$.
- Todo vértice $i \in N$ com oferta $b(i)$ terá oferta igual a $b(i) + x(\delta^-(i)) - x(\delta^+(i))$ em $G(x)$.

Notações e Suposições

- Para todo i em N representamos por $\pi(i)$ a variável dual que representa o potencial de i.
- Para toda aresta (i, j) em A ou $A(x)$ definimos o custo reduzido $c_{ij}^{\pi} = c_{ij} - \pi(i) + \pi(j)$.

Otimalidade Aproximada

- Dado um fluxo ou pseudo-fluxo x e um $\epsilon > 0$ dizemos que x é ϵ-ótimo se existe π tal que o par (x, π) satisfaz:
  - se $c_{ij}^{\pi} > \epsilon$ então $x_{ij} = 0$.
  - se $-\epsilon \leq c_{ij}^{\pi} \leq \epsilon$ então $0 \leq x_{ij} \leq u_{ij}$.
  - se $c_{ij}^{\pi} < -\epsilon$ então $x_{ij} = u_{ij}$.
- Notem que quando $\epsilon = 0$ temos as condições de otimalidade.

Otimalidade Aproximada

- Em $G(x)$ as condições de ϵ-otimalidade são simplificadas, de modo que x é ϵ-ótimo se, para toda aresta (i, j) em $A(x)$, temos $c_{ij}^{\pi} \geq -\epsilon$.

- Vamos mostrar isto tomando uma aresta (i, j) em A e considerando os diferentes valores de fluxo que podem passar por ela.

- Se $x_{ij} = 0$ então temos de mostrar que $c_{ij}^{\pi} \geq -\epsilon$. Mas sabemos que (i, j) está em $A(x)$. Portanto $c_{ij}^{\pi} = c_{ij} - \pi(i) + \pi(j) \geq -\epsilon$.

Otimalidade Aproximada

- Se $x_{ij} = u_{ij}$ então temos de mostrar que $c_{ij}^{\pi} \leq \epsilon$. Mas sabemos que (j, i) está em $A(x)$. Portanto $c_{ji}^{\pi} = -c_{ij} - \pi(j) + \pi(i) \geq -\epsilon$. Logo $c_{ij}^{\pi} = c_{ij} - \pi(i) + \pi(j) \leq \epsilon$.

- Por fim, se $0 \leq x_{ij} \leq u_{ij}$ então temos de mostrar que $-\epsilon \leq c_{ij}^{\pi} \leq \epsilon$. Mas sabemos que (i, j) e (j, i) estão em $A(x)$. Portanto $c_{ij}^{\pi} \geq -\epsilon$ e $c_{ij}^{\pi} = -c_{ji}^{\pi} \leq \epsilon$. $\square$

Otimalidade Aproximada

Lema

Qualquer fluxo viável para o problema de fluxo de custo mínimo é ϵ-ótimo se $\epsilon \geq C$. Além disso, se $\epsilon < 1/n$ então qualquer fluxo factível ϵ-ótimo é de custo mínimo.

- Sendo x um fluxo viável e $\pi = 0$ temos $c_{ij}^{\pi} = c_{ij} \geq -C \geq -\epsilon$. Logo x é ϵ-ótimo.
- Agora consideramos um fluxo ϵ-ótimo x para $\epsilon < 1/n$ e um potencial π.

Otimalidade Aproximada

- Consideramos um circuito orientado W em $G(x)$.
- Para toda aresta $(i, j) \in A$ temos que $c_{ij}^{\pi} \geq -\epsilon > -1/n$.
- Logo $\sum_{(i,j)\in W} c_{ij}^{\pi} \geq -\epsilon n > -1$.
- Como os custos são inteiros temos que o custo reduzido do circuito é não negativo.

Otimalidade Aproximada

- Notem que:

$$\begin{aligned}\sum_{(i,j)\in W} c_{ij}^{\pi} &= \sum_{(i,j)\in W} (c_{ij} - \pi(i) + \pi(j)) \\ &= \sum_{(i,j)\in W} c_{ij}.\end{aligned}$$

- Logo $G(x)$ não tem circuitos de custo negativo o que implica que x é um fluxo de custo mínimo. □

Descrição do Algoritmo

Algoritmo *fluxo-mínimo-com-escala-nos-custos*

1. $\pi = 0$
2. $\epsilon = C$
3. Obtenha um fluxo viável x
4. **enquanto** $\epsilon \geq 1/n$ **faça**
    1. *melhore-aproximação*(ϵ, x, π)
    2. $\epsilon = \epsilon/2$
5. Devolva x.

Descrição do Algoritmo

- Dizemos que um vértice i é ativo se seu excesso $e(i) > 0$.
- Dizemos que uma aresta (i, j) é admissível se $0 > c_{ij}^{\pi} \geq -\epsilon/2$.

Descrição do Algoritmo

Algoritmo *melhore-aproximação*(ϵ, x, π)

1. **para** toda aresta $(i, j) \in A$ **faça**
   1. **se** $c_{ij}^{\pi} > 0$ **então** $x_{ij} = 0$
   2. **senão se** $c_{ij}^{\pi} < 0$ **então** $x_{ij} = u_{ij}$
2. calcule os excessos dos vértices
3. **enquanto** existe um vértice ativo **faça**
   1. pegue um vértice ativo i
   2. *envie/desloque*(i)

Descrição do Algoritmo

Algoritmo *envie/desloque*(i)

1. **se** $G(x)$ contém uma aresta admissível $(i, j) \in A$
2. **então** envie $\delta = \min\{e(i), r_{ij}\}$ unidades de fluxo de i para j.
3. **senão** $\pi(i) = \pi(i) + \epsilon/2$

Descrição do Algoritmo

- Dizemos que um envio através de uma aresta (i, j) é saturante se $\delta = r_{ij}$.
- Caso contrário dizemos que o envio é não saturante.

Análise do Algoritmo

Lema

A função melhore-aproximação mantém um pseudofluxo $\epsilon/2$-ótimo e termina com um fluxo $\epsilon/2$-ótimo.

- Vamos provar usando indução no número de envios e deslocamentos.
- No caso base analisamos o pseudofluxo inicial.
- Nele toda aresta com custo reduzido positivo tem fluxo igual a zero e toda aresta com custo reduzido negativo tem fluxo igual à sua capacidade.
- Logo temos um pseudofluxo 0-ótimo, logo ele também é $\epsilon/2$-ótimo.

Análise do Algoritmo

- Num envio através de uma aresta (i, j) sabemos que $c_{ij}^{\pi} < 0$ e podemos adicionar a aresta (j, i) a $G(x)$.

- Notem que:

$$\begin{aligned} c_{ji}^{\pi} &= -c_{ij} - \pi(j) + \pi(i) \\ &= -(c_{ij} - \pi(i) + \pi(j)) \\ &= -c_{ij}^{\pi} > 0. \end{aligned}$$

- Portanto (j, i) não viola a condição de $\epsilon/2$ otimalidade.

Análise do Algoritmo

- Num deslocamento de um vértice i sabemos que $c_{ij}^{\pi} \geq 0$ para toda aresta (i, j) saindo de i.
- Como $\pi(i)$ aumenta de $\epsilon/2$ temos que $c_{ij}^{\pi} = c_{ij} - \pi(i) + \pi(j)$ diminui de $\epsilon/2$.
- Como $c_{ij}^{\pi} \geq -\epsilon/2$ respeita a condição de $\epsilon/2$ otimalidade as arestas que saem de i não violam a condição.
- Quanto às arestas (j, i) chegando em i temos que $c_{ji}^{\pi} = c_{ji} - \pi(j) + \pi(i)$ aumenta de $\epsilon/2$.
- Portanto as arestas que chegam em i continuam respeitando a condição de $\epsilon/2$ otimalidade. □

Análise do Algoritmo

- Agora vamos analisar a complexidade de tempo do algoritmo.
- Para tanto vamos contabilizar o número de operações de deslocamentos, envios saturantes e envios não saturantes.

Análise do Algoritmo

Lema

O potencial de qualquer vértice sofre no máximo $3n$ deslocamentos durante uma execução da função melhore-aproximação.

- Seja x um pseudofluxo $\epsilon/2$-ótimo qualquer da chamada atual de *melhore-aproximação*.
- Seja x' o fluxo ϵ-ótimo devolvido pela chamada anterior de *melhore-aproximação*.
- Sejam π e π' os potenciais dos vértices correspondentes a x e x', respectivamente.

Análise do Algoritmo

- Vamos usar sem provar o fato de que, para todo vértice i^* com excesso existe um vértice j^* com déficit, para o qual existe um conjunto de vértices que forma um caminho P em $G(x)$ indo de i^* até j^* e um caminho P' em $G(x')$ indo de j^* até i^*.

Análise do Algoritmo

- Como x é um pseudofluxo $\epsilon/2$-ótimo para toda aresta (i,j) em $G(x)$ temos $c_{ij}^{\pi} \geq -\epsilon/2$. Logo $\sum_{(i,j)\in P} c_{ij}^{\pi} \geq -n\epsilon/2$.
- Sabendo que $c_{ij}^{\pi} = c_{ij} - \pi(i) + \pi(j)$ temos:

$$\begin{aligned} \sum_{(i,j)\in P} c_{ij}^{\pi} &= \sum_{(i,j)\in P} (c_{ij} - \pi(i) + \pi(j)) \\ &= \sum_{(i,j)\in P} c_{ij} - \pi(i^*) + \pi(j^*) \\ &\geq -n\epsilon/2. \end{aligned}$$

- Portanto:

$$\pi(i^*) - \sum_{(i,j)\in P} c_{ij} \leq n\epsilon/2 + \pi(j^*). \qquad (1)$$

Análise do Algoritmo

- De modo semelhante, como x' é um fluxo ϵ-ótimo para toda aresta (i,j) em $G(x')$ temos $c_{ij}^{\pi'} \geq -\epsilon$. Logo $\sum_{(j,i)\in P'} c_{ji}^{\pi'} \geq -n\epsilon$.

- Sabendo que $c_{ji}^{\pi'} = -c_{ij} - \pi'(j) + \pi'(i)$ temos:

$$\begin{aligned}
\sum_{(j,i)\in P'} c_{ji}^{\pi'} &= \sum_{(j,i)\in P'} (-c_{ij} - \pi'(j) + \pi'(i)) \\
&= -\sum_{(j,i)\in P'} c_{ij} - \pi'(j^*) + \pi'(i^*) \\
&\geq -n\epsilon.
\end{aligned}$$

- Portanto:

$$\sum_{(j,i)\in P'} c_{ij} - \pi'(i^*) \leq n\epsilon - \pi'(j^*). \qquad (2)$$

Análise do Algoritmo

- Somando as inequações (1) e (2) temos $\pi(i^*) - \pi'(i^*) \leq 3n\epsilon/2 + \pi(j^*) - \pi'(j^*)$.
- Como, por hipótese, $j*$ tem excesso negativo ele nunca é escolhido para uma operação de deslocamento.
- Logo $\pi(j^*) = \pi'(j^*)$ e $\pi(i^*) - \pi'(i^*) \leq 3n\epsilon/2$.
- Mas, em cada operação de deslocamento o potencial do vértice é aumentado de $\epsilon/2$.
- Logo o número de operações de deslocamento sofridas por um vértice numa execução da função *melhore-aproximação* é no máximo $3n$. □

Análise do Algoritmo

Lema

A função melhore-aproximação realiza $O(nm)$ envios saturantes.

- Vamos limitar o número de envios saturantes por aresta mostrando que entre dois envios saturantes na mesma aresta é necessário que os dois extremos desta sofram operações de deslocamento.

- Para tanto basta tomar uma aresta (i, j) e considerar que esta sofreu um envio saturante. Neste caso sabemos que $c_{ij}^{\pi} < 0$.

Análise do Algoritmo

- Antes que possa ser enviado mais fluxo através de (i, j) deve ser enviado fluxo através de (j, i). Para tanto é necessário que $c_{ji}^{\pi} < 0$.

- Neste caso temos:

$$\begin{aligned} c_{ij}^{\pi} &= c_{ij} - \pi(i) + \pi(j) \\ &= -c_{ji} + \pi(j) - \pi(i) \\ &= -c_{ji}^{\pi} > 0. \end{aligned}$$

- Isto só acontece se j for deslocado.

Análise do Algoritmo

- De modo semelhante, para voltar a enviar fluxo por (i, j) é necessário que $c_{ij}^{\pi} < 0$.
- Para tanto o vértice i precisa ser deslocado.
- Mas o lema anterior limitou em $O(n)$ o número de deslocamentos que cada vértice sofre numa execução de *melhore-aproximação*.
- Logo o número total de envios saturantes é $O(nm)$. □

Análise do Algoritmo

- Agora resta limitar o número de envios não saturantes.
- Para tanto definimos o grafo admissível $G'(x)$, que corresponde ao grafo residual $G(x)$ restrito às arestas (i, j) admissíveis, ou seja, com $0 > c_{ij}^{\pi} \geq -\epsilon/2$.

Análise do Algoritmo

Lema

O grafo admissível $G'(x)$ é sempre acíclico ao longo da execução de melhore-aproximação.

- Vamos provar usando indução no número de envios e deslocamentos.
- No caso base temos que o grafo admissível inicial não possui arestas.
- Isto vale porque toda aresta (i, j) com $c_{ij}^{\pi} < 0$ tem fluxo $x_{ij} = u_{ij}$ e, portanto, não está no grafo residual $G(x)$.
- Assim $G'(x)$ inicial é trivialmente acíclico.

Análise do Algoritmo

- Num envio através de uma aresta (i, j) é possível que sua inversa (j, i) seja adicionada ao grafo residual $G(x)$.

- Mas sabemos que (i, j) é admissível, ou seja, $c_{ij}^{\pi} < 0$. Logo:

$$\begin{aligned} c_{ji}^{\pi} &= -c_{ij} - \pi(j) + \pi(i) \\ &= -(c_{ij} - \pi(i) + \pi(j)) \\ &= -c_{ij}^{\pi} \\ &> 0. \end{aligned}$$

- Portanto (j, i) não é adicionada no grafo admissível $G'(x)$. Logo envios não criam circuitos em $G'(x)$.

Análise do Algoritmo

- Numa operação de deslocamento em um vértice ativo i temos que $\pi(i)$ aumenta de $\epsilon/2$.
- Se i sofre um deslocamento é porque toda aresta (i, j) que sai de i tem custo reduzido $c_{ij}^{\pi} \geq 0$.
- Depois do deslocamento estas arestas tem seu custo reduzido $c_{ij}^{\pi} = c_{ij} - \pi(i) + \pi(j)$ diminuído de $\epsilon/2$ podendo ser acrescentadas em $G'(x)$.

Análise do Algoritmo

- Mas, toda aresta (j, i) incidente em i tem seu custo reduzido $c_{ji}^{\pi} = c_{ji} - \pi(j) + \pi(i)$ aumentado de $\epsilon/2$.
- Para toda aresta (j, i) em $G(x)$ vale que $c(j, i) \geq -\epsilon/2$.
- Assim, depois do aumento em $\pi(i)$ as arestas incidentes em i se tornam inadimissíveis.
- Logo a operação de deslocamento não pode produzir circuitos e o resultado segue. $\square$

Análise do Algoritmo

Lema

A função melhore-aproximação realiza $O(n^2 m)$ envios não saturantes.

- Nesta prova vamos utilizar uma função potencial Φ.
- Primeiro definimos, para todo vértice i, a função $g(i)$ como sendo o número de vértices alcançáveis a partir de i no grafo admissível $G'(x)$.
- Nesta definição consideramos que todo vértice alcança a si mesmo, assim $g(i) \geq 1$ para todo vértice i.

Análise do Algoritmo

- Definimos nossa função potencial

$$\Phi = \sum_{i \ \ ativos} g(i).$$

- Uma observação importante para a prova é que $\Phi \geq 0$.

Análise do Algoritmo

- Vamos limitar o valor que Φ pode alcançar ao longo de uma execução de *melhore-aproximação*.
- Primeiro limitamos o valor inicial de Φ.
- Como no inicio o grafo admissível $G'(x)$ não possui arestas, para todo vértice i temos $g(i) = 1$.
- Portanto o valor inicial de $\Phi \leq n$.

Análise do Algoritmo

- Vamos limitar o aumento em Φ decorrente de envios saturantes.
- Quando ocorre um envio saturante numa aresta (i, j) o vértice j pode se tornar ativo.
- Com isso Φ pode aumentar de $g(j) \leq n$.
- Como o número de envios saturantes é $O(nm)$ temos que o aumento em Φ por conta destes envios é $O(n^2 m)$.

Análise do Algoritmo

- Vamos limitar o aumento em Φ decorrente de deslocamentos.
- Quando ocorre um deslocamento num vértice i podem surgir novas arestas admissíveis incidentes do tipo (i, j).
- Com isso i pode passar a alcançar novos vértices fazendo com que $g(i)$ aumente de no máximo n.

Análise do Algoritmo

- Por outro lado o deslocamento torna todas as arestas do tipo (j, i) que chegam em i inadimissíveis.
- Por isso para qualquer $k \neq i$ temos que $g(k)$ não se altera.
- Assim, o aumento em Φ por deslocamento é de no máximo n.
- Como o número de deslocamentos por vértice numa execução de *melhore-aproximação* é $O(n)$ temos que o aumento total de Φ decorrente de deslocamentos é $O(n^3)$.

Análise do Algoritmo

- Por fim vamos analisar a variação em Φ decorrente de envios não saturantes.
- Quando ocorre um envio não saturante numa aresta (i, j) o vértice j pode se tornar ativo e, com certeza, o vértice i se torna inativo.
- Podemos perceber que todo vértice alcançável a partir de j também é alcançável a partir de i, pois (i, j) é admissível.

Análise do Algoritmo

- No entanto, i não é alcançável a partir de j, pois $G'(x)$ é acíclica.
- Portanto $g(j) \leq g(i) - 1$.
- Logo, para cada envio não saturante Φ diminui de pelo menos uma unidade.
- Como Φ é não negativa, o número de envios não saturantes é limitado pelo valor máximo que Φ pode alcançar, que é $O(n) + O(n^2m) + O(n^3) = O(n^2m)$. $\square$

Análise do Algoritmo

Teorema

O algoritmo de fluxo de custo mínimo com escala nos custos tem complexidade de tempo $O(n^2 m \log(nC))$.

- Primeiro vamos analisar a complexidade de tempo da função *melhore-aproximação*.
- Ao longo da execução da função encontrar arestas admissíveis leva tempo $O(nm)$ e o custo para encontrar vértices ativos pode ser embutido neste custo.

Análise do Algoritmo

- Sabemos que as operações de envio e deslocamento tem complexidade de tempo $O(1)$.
- Portanto, o custo total dos deslocamentos é $O(n^2)$, dos envios saturantes é $O(nm)$ e dos envios não saturantes é $O(n^2m)$.
- Assim a complexidade de tempo da função *melhore-aproximação* é $O(n^2m)$.

Análise do Algoritmo

- A função principal do algoritmo executa $1 + \lceil log(nC) \rceil$ chamadas por *melhore-aproximação*.
- Assim chegamos à complexidade de tempo $O(n^2 m \log(nC))$ enunciada no lema. □
- De modo semelhante ao que acontece com o algoritmo de préfluxo, a complexidade de tempo deste algoritmo pode ser melhorada através de escolhas apropriadas dos vértices ativos.

Referências

Ravindra K. Ahuja, Thomas L. Magnanti e James B. Orlin.
Network Flows: Theory, Algorithms and Applications..
Prentice-Hall, 1993.