Novo teste de redução para o problema da árvore de Steiner com coleta de prêmios*

Gabriel M. de Azevedo[1], **Carlos E. Ferreira**[1]

[1]Departamento de Ciência da Computação – Universidade de São Paulo (USP)
Rua do Matão – 1010 – São Paulo – SP – Brazil

{gabrielmorete,cef}@ime.usp.br

Abstract. *The prize-collecting Steiner tree problem asks for a subgraph minimizing the sum of the costs of all edges of the subgraph plus the total cost of all vertices not contained in the subgraph. We present a new reduction test for the problem and computational results with instances from the literature.*

Resumo. *O problema da árvore de Steiner com coleta de prêmios consiste na busca de um subgrafo que minimiza a soma dos valores das arestas contidas no subgrafo e dos vértices não contidos. Apresentamos um novo teste de redução para o problema e resultados computacionais com instâncias da literatura.*

1. Introdução

Dados um grafo $G = (V, E)$, uma função $p : V \to \mathbb{Q}_{\geq}$ e uma função $c : E \to \mathbb{Q}_{\geq}$ o problema da árvore de Steiner com coleta de prêmios consiste em encontrar um subgrafo conexo $T \subseteq G$ que minimiza a soma dos valores das arestas contidas no subgrafo e a soma dos valores do vértices que não estão contidos, ao longo do texto denotaremos esse valor por $\mathrm{GW}(T) = c(E(T)) + p(V(G) \setminus V(T))$. Apesar de não exigirmos que a solução seja uma árvore, sempre podemos construir uma árvore de mesmo custo partindo de uma solução ótima. Definimos o conjunto de vértices consumidores como $R = \{v \in V : p_v > 0\}$, os demais vértices serão chamados de não-consumidores. O problema é $\mathcal{NP}$-difícil pois é uma generalização do problema da árvore de Steiner, que é $\mathcal{NP}$-difícil [Karp 1972].

O problema possui diversas aplicações práticas, uma aplicação proposta em [Ljubić et al. 2005] é a expansão de redes de fibra ótica. O valor de cada vértice é o lucro de conectar uma residência, vértices não-consumidores são intersecções de ruas e o valor de cada aresta é o custo de criar a infraestrutura em cada rua.

Por sua vez, um teste de redução é um procedimento que transforma uma instância de um problema em uma outra instância equivalente de menor tamanho, isto é, tal que existe uma função que mapeia uma solução ótima da instância reduzida em uma solução ótima da instância original.

Devido às aplicações, há um grande empenho na formulação de algoritmos exatos para o problema, em geral, algoritmos *branch and cut*. Porém, pela natureza difícil do problema, é desejável aplicar testes de redução, visando diminuir o tamanho das instâncias para viabilizar a utilização de algoritmos exatos. Diversos testes efetivos são propostos em [Uchoa 2006] e [Rehfeldt and Koch 2020].

*Pesquisa financiada pela FAPESP processo número 20/10341-3.

A contribuição deste texto é a apresentação de um novo teste de redução. O teste proposto é uma modificação do *Minimum Adjacency Test* (ou $V \backslash K$ *reduction test*) [Ljubić et al. 2005]. O teste original é aplicado em algoritmos *branch and cut* baseados na redução ao problema da arborescência de Steiner, mas, se aplicado diretamente a uma instância, pode modificar a solução ótima.

2. O teste de redução da árvore geradora mínima

Definimos a contração de uma aresta uv de G como a formação de um novo grafo $G' = G \setminus \{u, v\} \cup \{w\}$ tal que a $wx \in E(G')$ se só se vx ou $ux \in E(G)$, em caso de arestas múltiplas mantemos a de menor custo. Vamos definir e demonstrar o teste original.

Teorema 1 (Minimum Adjacency Test)**.** *Se há dois vértices adjacentes $v, u \in R$ tais que*

$$\min\{p_v, p_u\} > c_{vu} \quad \textit{e} \quad c_{vu} = \min_{vw \in E} c_{vw}$$

então, se ao menos um dos vértices está em uma solução viável, existe uma solução viável contendo ambos os vértices, a aresta uv e com custo limitado pela solução atual.

Demonstração. Considere, sem perda de generalidade, uma solução viável T tal que $v \in V(T)$ e $u \notin V(T)$. Então, $T' = (V(T) \cup \{u\}, E(T) \cup \{uv\})$ é uma solução viável com $\mathrm{GW}(T') = \mathrm{GW}(T) + c_{vu} - p_u < \mathrm{GW}(T) + c_{vu} - c_{vu} < \mathrm{GW}(T)$. Portanto, encontramos uma solução de custo inferior.

Agora, considere uma solução viável T tal que $v, u \in V(T)$ e $vu \notin E(T)$. Considere o caminho $P = (v, e_1, \ldots, e_k, u)$, como por hipótese $c_{vu} = \min_{vw \in E} c_{vw}$ então $c_{vu} \leq c_{e_1}$ e geramos a solução viável $T' = (V(T), E(T) \cup \{uv\} \setminus \{e_1\})$ (basta observar que uv mantém a conexidade) com $\mathrm{GW}(T') = \mathrm{GW}(T) + c_{uv} - c_{e_1} \leq \mathrm{GW}(T)$. □

O teste da literatura contrai as arestas que satisfazem a condição do teorema 1 durante a redução ao problema da arborescência de Steiner, gerando um novo vértice de custo $p_u + p_v - c_{uv}$. Note que, ao aplicar esse teste, possivelmente não geramos uma instância equivalente, pois, caso o vértice contraído não esteja na solução final, o valor ótimo pode ser alterado. O novo teste é uma modificação do teste acima que produz uma instância equivalente pois só retira arestas. Vamos definir um algoritmo que executa contrações sucessivas em um grafo. Ao executar uma contração, o algoritmo mantém o mesmo lucro para todos os vértices não contraídos e o mesmo custo para as arestas não contraídas. O vértice contraído recebe a soma do lucro dos vértices originais menos o custo da aresta contraída. Temos o seguinte algoritmo.

Algoritmo 1: Redução(G, p, c)

1. $L \leftarrow \emptyset$
2. **enquanto** *há $e \in E(G)$ que satisfaz as condições de custo do teorema 1* **faça**
3. $\quad L \leftarrow L \cup \{e\}$
4. $\quad G \leftarrow contrai(G, p, c, e)$
5. **retorna** L

Lema 1. *Seja $L \subseteq E(G)$ a lista de arestas devolvida pelo algoritmo 1 e seja K uma componente conexa do subgrafo induzido $G[L]$. Então, se há solução ótima que contém algum vértice de K, há solução ótima que contém K como um subgrafo.*

Demonstração. Primeiramente, se $L = \emptyset$ está feito pois a lista induz o grafo vazio. Vamos manter em L somente as arestas contidas na componente K, preservando a ordem de inserção. Como ao inserir uma aresta em L seus extremos são contraídos, K é uma árvore. Vamos provar por indução que se há uma solução ótima T^* que contém algum vértice de K então $V(K) \subseteq V(T^*)$. Se K contém um único vértice está feito por hipótese.

Considere K não trivial, seja v_iv_j a última aresta de L por ordem de inserção e seja K_i e K_j as duas componentes conexas de $K - \{v_iv_j\}$. Se ambas as componentes contém algum vértice de $V(T^*)$ está feito por hipótese de indução, então, suponha que não. Por hipótese do enunciado ao menos uma dessas componentes contém algum vértice de T^*, suponha que seja K_i, por hipótese de indução $V(K_i) \subseteq V(T^*)$. Como o algoritmo 1 contraiu a aresta v_iv_j, o vértice contraído w correspondente a componente K_j possui $p(w) = p(V(K_j)) - c(E(K_j)) > c_{v_iv_j}$. Considere o seguinte subgrafo T' tal que

$$T' = (V(T^*) \cup V(K_j), E(T^*) \cup E(K_j) \cup \{v_iv_j\})$$
$$\mathrm{GW}(T') = \mathrm{GW}(T^*) - p(V(K_j)) + c(E(K_j)) + c_{v_iv_j} < \mathrm{GW}(T^*)$$

como K_j é conexo, T' é uma solução viável de custo menor que T^*, contradição. Portanto, $V(K_j) \subseteq V(T^*)$ e $V(K) \subseteq V(T^*)$.

Agora, vamos provar que existe uma solução ótima que contém L. Suponha que não há, tome T^* uma solução ótima que contém o maior prefixo de L em ordem de contração. Seja G' o grafo gerado após contrair esse prefixo em ordem e $v_iv_j \in E(G')$ a aresta que seria contraída nessa etapa ($v_iv_j \notin E(T^*)$). Considere o caminho $P = (v_i, e_1, \ldots, e_l, v_j)$ em G' induzido pelas arestas de T^* (sabemos que existe pois T^* é conexo e contém $V(K)$). Pela hipótese do teorema 1, vale que v_iv_j tem custo mínimo na adjacência de v_j. Podemos tomar $T' = (V(T^*), E(T^*) \cup \{v_iv_j\} \setminus \{e_l\})$, que mantém a conexidade e possui custo limitado por T^*. Porém, T' contém um prefixo de L maior, uma contradição. Portanto, há solução ótima que contém K como subgrafo. □

Teorema 2. *Sejam $K_1, \ldots, K_l$ as componentes conexas induzidas pelas arestas escolhidas pelo algoritmo 1. Então podemos manter somente uma aresta de custo mínimo no corte $\delta_G(K_i, K_j)$ que liga as componentes.*

Demonstração. Tome a lista devolvida pelo algoritmo 1. Aplicando o lema 1 para as partes disjuntas de cada componente, existe uma solução ótima em que essas componentes são subgrafos ou estão ausentes. Com isso, caso duas componentes estejam presentes, haverá no máximo uma aresta cruzando esse corte, ou, do contrário, teríamos um circuito. Portanto, podemos manter somente uma aresta de custo mínimo cruzando o corte. □

3. Testes computacionais

O teorema 2 nos da uma condição para retirar arestas sem modificar uma solução ótima. A implementação utiliza *Union find* para realizar uma pseudo-contração. Percorremos todas as arestas até que não haja contração possível resultando na complexidade $O(|V||E|)$. Realizamos experimentos computacionais em instâncias da literatura disponíveis na página do [DIMACS 2014], o tamanho original e o percentual reduzido de algumas instâncias estão exibidos na tabela 1. Também exibimos os resultados obtidos pelo *least cost test* [Uchoa 2006, Ljubić et al. 2005], um teste de redução muito utilizado.

Os experimentos revelam que o teste é efetivo em instâncias densas, reduzindo até

47% das arestas de instâncias dos conjuntos C, D, E e até 36% nas instâncias do conjunto $RANDOM$. Porém, esse valor reduz para 3% ou menos em instâncias esparsas. O tempo consumido não foi exibido pois o processo consumiu frações de segundos em todas as instâncias, o que é uma vantagem em relação ao *least cost test* pois o mesmo precisa calcular a distância entre todos os pares de vértices adjacentes.

Nome	Arestas	Redução		Nome	Arestas	Redução	
		lct	novo			lct	novo
C19-A	12500	38.69%	16.43%	D19-B	25000	37.26%	8.62%
C20-A	12500	38.47%	43.91%	D20-B	25000	36.86%	46.67%
C19-B	12500	38.69%	19.05%	E19-A	62500	23.32%	13.51%
C20-B	12500	38.47%	47.08%	E20-A	62500	23.42%	43.97%
D19-A	25000	37.26%	4.30%	E19-B	62500	23.32%	17.98%
D20-A	25000	36.86%	43.53%	E20-B	62500	23.42%	46.73%
a1000.3	8107	5.73%	24.42%	a14000.3	111869	0.54%	26.69%
a1800.3	14531	3.70%	27.07%	a4000.3	32025	1.87%	26.89%
a0200.3	1616	15.16%	36.57%	a0800.3	6385	6.81%	25.47%
a12000.3	96449	0.64%	26.08%	a1400.3	11263	4.60%	22.29%
a2000.3	15751	3.08%	24.35%	a6000.3	47915	1.22%	24.49%
a0400.3	3222	11.01%	29.11%	a10000.3	79778	0.77%	27.36%
a1200.3	9451	4.83%	25.98%	a1600.3	12963	3.66%	26.81%
a3000.3	24026	2.36%	27.33%	a8000.3	64177	0.95%	26.25%
a0600.3	4808	7.94%	25.54%	drosophila75	93394	0%	4.42%

Tabela 1. Redução para as instâncias C, D, E, ACTMODPC, RANDOM

4. Conclusão

O novo teste de redução é uma extensão interessante de um teste já utilizado na literatura. Testamos sua eficácia em instâncias da literatura, comparando com o teste de redução *least cost test*, que é muito utilizado. Podemos ver que na maior parte das instâncias densas o novo teste mostrou melhores resultados.

Referências

DIMACS (2014). 11th dimacs implementation challenge. `dimacs11.zib.de`. Acessado em 21/06/2020.

Karp, R. (1972). Reducibility among combinatorial problems. *Complexity of Computer Computations*, pages 85–103.

Ljubić, I., Weiskircher, R., Pferschy, U., Klau, G., Mutzel, P., and Fischetti, M. (2005). An algorithmic framework for the exact solution of the prize-collecting steiner tree problem. *Mathematical Programming*, 105(2-3):427–449.

Rehfeldt, D. and Koch, T. (2020). On the exact solution of prize-collecting steiner tree problems. Technical report, Zuse Institute Berlin.

Uchoa, E. (2006). Reduction tests for the prize-collecting steiner problem. *Operations Research Letters*, 34(4):437–444.