Problema da Floresta de Steiner em Grafos- Estudo Orientado 2019.2

Sérgio de Melo Barreto Junior

09 de Dezembro de 2019

1 Introdução

O Problema da Floresta de Steiner (SFP) é uma generalização natural do Problema da Árvore de Steiner. A tarefa referente ao SFP é conectar várias redes de terminais. Dado um grafo não orientado G = (V, E) e um conjunto de subconjuntos disjuntos de V , T_1, T_2, ... T_t, pretende-se determinar um sub-grafo de G com custo mínimo, em que cada par de nós de cada T_k estejam ligados. Como SFP é uma generalização do problema da árvore de Steiner, portanto, é NP-difícil.[1, 2, 3]

Com seu caráter NP-difícil, alguns algoritmos aproximativos são desenvolvidos para se buscar o melhor resultado. O algoritmo clássico de Agrawal, Klein e Ravi [4], declarado no cenário do esquema primal-duplo de Goemans e Williamson [5] utiliza o relaxamento de corte não direcionado para o problema da floresta Steiner. Sua taxa de aproximação é 2 - $\frac{1}{k}$, onde k é o número de pares de terminais.

Outro algoritmo aproximativo utiliza heurísticas propostas para o problema de Steiner. Dada uma permutação i_1, i_2, . . . , i_t, a heurística Tipo a Tipo calcula na interação k uma solução de Steiner para o grafo $<V_{i_k}>$, sub-grafo de G induzido por V_{i_k}, atualizando os custos dos nós v da solução encontrada fazendo o custo das arestas $c_v = 0$, uma vez que já estando na solução não há custo adicional por serem reutilizados. Em seguida, a próxima interação é executada. A solução final é encontrada tornando minimal a união dos nós das soluções de Steiner obtidas em cada interação.[6]

Quando analisamos outras problemas de otimização, encontramos versões híbridas de metaheurísticas que incorporam técnicas de mineração de dados a diversos problemas de otimização. A heurística DM - GRASP é um desses métodos híbridos que alcançou resultados promissores. É uma versão híbrida da metaheurística GRASP que incorpora um processo de mineração de dados. Os resultados dos experimentos demonstraram que a incorporação do procedimento de mineração de dados no GRASP original melhorou não apenas a eficácia do método, em termos de qualidade das soluções obtidas, mas também sua eficiência, em termos do tempo necessário para alcançar uma solução alvo. [7, 8]

2 Objetivo

Dada a importância do problema da Floresta de Steiner e a oportunidade apresentada pela união de técnicas de mineração de dados com otimização, o objetivo do estudo orientado é contribuir na construção de uma versão híbrida de heurística que incorpore técnicas de mineração de dados para o problema da floresta de Steiner. Nesse primeiro momento o objetivo é construir uma heurística para o problema da Floresta de Steiner por meio da redução do problema a um conjunto de problemas da árvore de steiner. Diferentemente do método apresentado por Brás [6], que segue um modelo incremental de construção da solução final, o método proposto divide o problema em subproblemas independentes da árvore de Steiner.

3 Heurística proposta

Com o objetivo de desenvolver uma heurística para a Floresta de Steiner por meio da redução do problema a diversos problemas da Árvore de Steiner em grafos, desenvolveu-se o seguinte pseudo código:

Input: um grafo não orientado G = (V, E) e k conjunto de subconjuntos disjuntos de V , T_1, T_2, ... T_k;
Output: um sub-grafo de G com custo mínimo, em que cada par de nós de cada T_k estejam ligados;

Heurística Floresta de Steiner();
$G_t \leftarrow \emptyset$;
$G_{Final} \leftarrow \emptyset$;
$G_{indu} \leftarrow \emptyset$;
for *i:0 to k* **do**
  $G_s \leftarrow \emptyset$;
  $G_s \leftarrow$ Steiner Solution (G, G[i] terminais);
  $G_t \leftarrow G_t \cup G_s$;
end
for ***each*** *Connected Component G' of* G_t **do**
  $G_{Final} \leftarrow \text{Prim}(G_{indu})$;
end
return G_{Final};

Algorithm 1: Algoritmo Floresta de Steiner

O algoritmo proposto calcula para cada subconjunto disjunto T_k a solução da árvore de steiner, considerando todos os outros nós, inclusive os nós dos demais conjuntos de terminais, como possíveis conexões para formar a árvore referente a T_k. A solução final do algoritmo é definida unindo as k árvores resultantes em um grafo e eliminando para cada componente conexa resultante, caso haja, os cíclos por meio da algoritmo de Prim sob o componente conexo.

Para identificação das componentes conexas, foi utilizado o seguinte algo-

ritmo:

1) Initialize all vertices as not visited
2) Do following for every vertex 'v'.

   (a) If 'v' is not visited before, call DFSUtil(v)

DFSUtil(v)

1) Mark 'v' as visited
2) Do following for every adjacent 'u' of 'v';

   (a) If 'u' is not visited, then recursively call DFSUtil(u)

4 Resultados Experimentais

Para análise do algoritmo proposto realizou-se experimentos sob instâncias do problema da Floresta de Steiner.

4.1 Instâncias

As instâncias utilizadas para os experimentos estão separadas em dois grupos compostos com diferentes configurações:

	Opções
Vértices	25/ 50/ 100/ 200/ 500
Arestas	160/ 200
Conjunto de terminais	2/ 3/ 4/ 5/ 10/ 15/ 20/ 35/ 50

Table 1: Conjunto JMP de instâncias

	Opções
Vértices	10/ 15/ 20
Arestas	20/ 30/ 45/ 50/ 100/ 105/ 190
Conjunto de terminais	3/ 4/ 5/ 6

Table 2: Conjunto MR de instâncias

E organizadas da seguinte forma em cada arquivo:

```
N 10
E 9 2 15
E 2 0 42
E 0 7 48
E 7 3 40
E 3 6 10
```

E 6 5 35
E 5 8 49
E 8 4 16
E 4 1 12
E 4 5 10
E 1 7 49
E 8 9 43
E 3 9 30
E 1 5 49
E 3 5 10
E 0 3 15
E 4 6 38
E 2 3 15
E 0 6 19
E 2 5 21
S 9 2 8 7
S 3 5
S 4 1 0 6

Primeira linha informando quantos vértices existem na instância, "E" indicando cada aresta ligando os dois inteiros consecutivos e com o custo referente ao último inteiro de cada linha. Ao final do arquivo, a letra "S" denota um conjunto de terminais.

Além disso, foi fornecido também uma tabela com a solução ótima de cada instância.

4.2 Experimentos

Com intuíto de analisar a heurística proposta, selecionou-se as seguintes instâncias do conjunto de dados JMP:

Inst.	Vertices	Conexões	Terminais	Sol.Ótima
b4781	50	200	5	89
b4786	100	200	5	350
b4791	100	200	10	359
b4911	100	200	15	480
b5192	500	200	35	3587.5
b5226	500	200	50	5285

Table 3: Instâncias de aplicação

O algoritmo de solução das árvores foi realizados com o pacote desenvolvido por Pajor, Wernek e Uchoa [9] com 100 iterações e semente 1. Para fins de análise, foram analisadas a qualidades e a eficiência do algoritmo proposto.

4.3 Resultado

Abaixo seguem os resultados do algoritmo:

Inst.	Vertices	Conexões	Terminais	heurística	tempo (s)	Sol.Ótima	% sol. Ótima
b4781	50	200	5	122	0.20	89	1.37
b4786	100	200	5	531	0.77	350	1.51
b4791	100	200	10	377	0.59	359	1.05
b4911	100	200	15	728	1.49	480	1.51
b5192	500	200	35	8041	38.25	3587	2.24
b5226	500	200	50	8489	95.69	5285	1.60

Table 4: Resultados do Algoritmo

O algoritmo proposto apresenta nas instâncias do experimento uma razão de aproximação média de 1.54 com uma variação de 1.05 até 2.24. Se compararmos com a taxa de aproximação do algoritmo proposto por Agrawal, Klein e Ravi [4], com taxa de aproximação de 2 - $\frac{1}{k}$, onde k é o número de pares de terminais, o algoritmo apresenta potencial em termos de qualidade da solução gerada, necessitando ainda uma comprovação formal da razão de aproximação.

5 Futuras evoluções

Com o algoritmo apresentado, é possível encontrar soluções para a Floresta de Steiner por meio da redução do problema a subproblemas independentes da árvore de Steiner, porém algumas evoluções podem, ainda, ser implementada em termos de eficiência e qualidade. Como subdividimos o problema em subproblemas independentes, há oportunidade da incremento na eficiência do algoritmo com a paralelização da etapa de construção das árvores de steiner para os subconjuntos disjuntos dos terminais e da etapa de eliminação de cíclos nos componentes conexos. Com isso, a etapa mais complexa poderá ser acelerada. Além disso, pode-se analisar o desempenho do algoritmo porposto com meta-heurísticas mais robustas da árvore de Steiner, o que gerará árvores melhores e consequentemente uma floresta melhor também.

Outra hipótese levantada é a possibilidade de melhoria na qualidade da solução final da floresta com a construção de uma nova árvore sob o grafo induzido no grafo original pelos vértices da componente conexa com cíclos. Acredita-se que com essa estratégia o algoritmo possa se aproveitar de conexões não consideradas anteriormente.

6 Referências

[1] Elisabeth Gassner, The Steiner Forest Problem revisited, Journal of Discrete Algorithms, v.8 n.2, p.154-163, June, 2010

[2] H.J. Promel, A. Steger, The Steiner Tree Problem, Advanced Lectures in Mathematics, Friedr. Vieweg Sohn, Braunschweig, 2002.

[3] Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-completeness. Freeman, San Francisco (1979)

[4] A. Agrawal, P. Klein, and R. Ravi. When trees collide: An approximation algorithm for the generalized Steiner problem on networks. SIAM Journal on Computing, 24(3):440–456, 1995.

[5] Goemans, M.X., Williamson, D.P.: A general approximation technique for constrained forest problems. SIAM Journal on Computing 24 (1995) 296–317

[6] BRÁS, Raúl Massano. Uma variante do problema da floresta de Steiner em grafos com aplicações em biologia da conservação. 2013.

[7] Santos, L. F.; Martins, S. L.; Plastino, A. Applications of the DM-GRASP heuristic: a survey. International Transactions in Operational Research 15 , 4 (2008), 387416.

[8] Rodrigues de Holanda Maia, Marcelo ; Plastino, Alexandre ; Vaz Penna, Puca Huachi . Hybrid data mining heuristics for the heterogeneous fleet vehicle routing problem. RAIRO-OPERATIONS RESEARCH, v. 52, p. 661-690, 2018.

[9] PAJOR, Thomas; UCHOA, Eduardo; WERNECK, Renato F. A robust and scalable algorithm for the Steiner problem in graphs. Mathematical Programming Computation, v. 10, n. 1, p. 69-118, 2018.