Árvore de Steiner - Relatório

Gabriel Medrano Silva

December 2023

1 O problema

O problema que foi alvo de estudo foi o problema da árvore de Steiner. Nesse problema, recebemos um grafo G e um conjunto de nós terminais Z. Queremos achar uma árvore que conecte todos os terminais e que tenha custo mínimo. Esse problema pode ser modelado como um problema de programação linear inteira da seguinte maneira:

$$\min \quad \sum_{e \in A} c_e x_e \tag{1}$$

$$x(\delta(W)) \geq 1, \qquad \forall W \subseteq V, \emptyset \neq W \cap Z \neq Z \tag{2}$$

$$x_e \in \{0, 1\}, \qquad \forall e \in A. \tag{3}$$

2 Implementação

Na parte de implementação, foi usado o software Gurobi 10.0.3. Optei por utilizar duas variáveis para cada aresta, ou seja, se $(i, j) \in A$, então temos 2 variáveis, $x_{(i,j)}$ e $x_{(j,i)}$ e uma restrição que $x_{(i,j)} = x_{(j,i)}$. Essa decisão foi tomada pois facilitava a implementação. Para resolver o problema, foi implementado 2 tipos diferentes de cortes, sendo o primeiro deles o do tipo (2) e o segundo os cortes de partição de Steiner. Os cortes do tipo (2) eram gerados tanto nas soluções fracionarias quanto em soluções inteiras inviáveis. Já os cortes de partição de Steiner, eram gerados apenas nas soluções fracionárias. Além dos cortes, optei também por implementar uma heurística primal que consistia em usar o valor da solução fracionária para criar uma árvore geradora máxima e remover as folhas não terminais dessa folha.

O número de cortes do tipo (2) foi limitado, pois a sua criação envolvia calcular a árvore de Gomory-Hu, que acabou sendo muito custoso. Não havia limite para os cortes de partição de Steiner, pois nos testes eles se mostraram bem eficientes. Também para evitar que o modelo ficasse muito pesado, foi limitado o número de cortes de Steiner gerados por execução do algoritmo de Gomoroy-Hu, ou seja, não foram gerados todos os cortes possíveis, apenas 2.

Esse número de duas restrições geradas foi obtido através de testes, tanto de números constantes, quanto de número que dependiam do tamanho da entrada.

A parte da heurística primal foi importante para calcular bons valores de incumbentes, que era algo que apenas os cortes não estavam conseguindo fazer em alguns dos casos testes.

Ao invés de começarmos a otimizar sem nenhuma restrição, escolhi incluir as restrições de Steiner quando $|W| = 1$.

3 Resultados

Os casos testes foram tirados do site steinlib.zib.de e foram testados os conjuntos de teste B e C. O computador tinha um AMD Ryzen 5 5600 6-Core Processor 3.50 GHz e 16,0 GB de RAM e Python 3.9.7.

Resultados para o conjunto de teste B:

Nome	$\|V\|$	$\|E\|$	$\|T\|$	Tempo (s)	Incumbent	BestBD	Ótimo
b01	50	63	9	0.66	82	82	82
b02	50	63	13	0.77	83	83	83
b03	50	63	25	0.53	138	138	138
b04	50	100	9	5	59	59	59
b05	50	100	13	1.72	61	61	61
b06	50	100	25	15.19	122	122	122
b07	75	94	13	1.61	111	111	111
b08	75	94	19	1.34	104	104	104
b09	75	94	38	6.45	220	220	220
b10	75	150	13	4.66	86	86	86
b11	75	150	19	4.15	88	88	88
b12	75	150	38	29.79	174	174	174
b13	100	125	17	19.11	165	165	165
b14	100	125	25	40.04	235	235	235
b15	100	125	50	12.33	318	318	318
b16	100	200	17	36.38	127	127	127
b17	100	200	25	38.09	127	127	131
b18	100	200	50	92.29	218	218	218

Resultados para o conjunto de teste C:

Nome	$\|V\|$	$\|E\|$	$\|T\|$	Tempo	Incumbent	BestBD	Ótimo
c01	500	625	5	93.56	85	85	85
c02	500	625	10	432.18	144	144	144
c03	500	625	83	600.17	796	648	754
c04	500	625	125	604.19	1161	1022	1079
c05	500	625	250	600.27	1632	1482	1579
c06	500	1000	5	90.79	55	55	55
c07	500	1000	10	614.36	124	98	102
c08	500	1000	83	617.40	593	470	509
c09	500	1000	125	659.77	785	576	707
c10	500	1000	250	610.81	1207	996	1093
c11	500	2500	5	492.56	32	32	32
c12	500	2500	10	614.24	46	42	46
c13	500	2500	83	609.43	285	205	258
c14	500	2500	125	604.90	437	263	323
c15	500	2500	250	611.18	921	495	556
c16	500	12500	5	641.10	13	11	11
c17	500	12500	10	755.10	31	15	18
c18	500	12500	83	602.82	315	86	113
c19	500	12500	125	615.75	174	94	146
c20	500	12500	250	684.50	548	259	267

4 Análise

Para instâncias com baixo número de terminais, o algoritmo performa relativamente bem. No entanto, para instâncias com um número mais elevado de terminais, o algoritmo performa muito pior, chegando a estourar o limite de 10 minutos.

Um exemplo disso são as instâncias c03 e c11. Apesar da c11 ter consideravelmente mais arestas, ela foi resolvida em menos de 10 minutos, enquanto a c03 não conseguiu ser resolvida.

Além disso, o uso dos cortes de partição de Steiner foi muito importante, pois sem eles, era possível resolver apenas instâncias muito pequenas. Outro fator que ajudou na resolução de entradas maiores, foi a heurística primal, que ajudou a achar valores bons para o incumbente.

Para exemplificar a eficiência tanto dos cortes de partição de Steiner quanto da heurística primal, em testes relativamente pequenos, como o b06, sem nenhuma dessas abordagens, o tempo para resolver é superior a 8 minutos, enquanto com essas duas abordagens o tempo cai para 15 segundos.

Nota-se que o principal problema na maioria das soluções é a dificuldade de aumentar o lower bound por um valor significativo, já que depois de certo tempo ele demorava muito para aumentar.