Pesquisa Operacional

Romário Oliveira Pantaleão, 2018.2.

Problema da árvore de Steiner : Variação limitando roteadores e elos

Romário Pantaleão
rop@ic.ufal.br

Resumo

Neste trabalho são propostas duas meta-heurísticas GRASP baseadas nos algoritmos de Kruskal e Prim para criação de árvore geradora mínima , aplicado a variação do *Steiner Tree Problem*. A variação escolhida se trata de limitações de elos, que são nós que conectam dois vértices e roteadores que conectam mais de dois vértices. Além dos dois algoritmos propostos usando GRASP, também é proposto um pré-processamento que identifica alguns subgrafos que certamente não farão parte de nenhuma solução ótima.

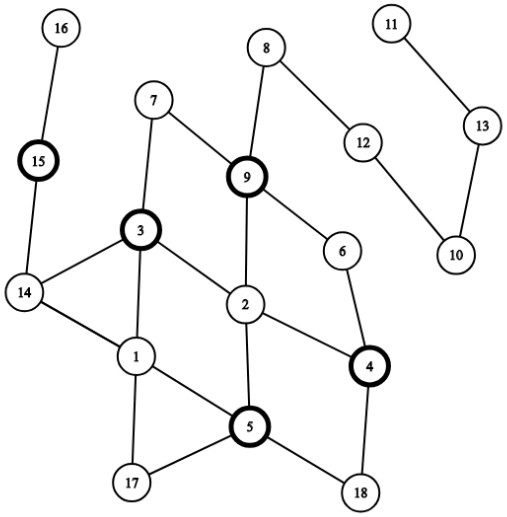
Com os resultados podemos observar uma pequena vantagem sobre a implementação baseada em Prim, porém a implementação usando Kruskal ainda pode ser certamente otimizada num trabalho futuro.

Palavras-chave: Steiner, Kruskal, LaTeX, Prim.

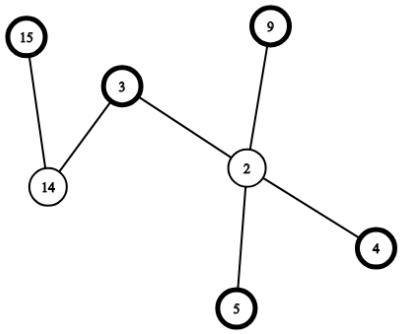
1 Introdução

O *Problema da Árvore de Steiner* pode ser considerado uma variação do *Problema da Árvore Geradora Mínima*, só que nesse caso do nosso trabalho proposto, não estamos preocupados com os pesos das arestas, e também não estamos preocupados em conectar todos os vértices existentes no grafo. No *Problema da Árvore de Steiner* nós estamos preocupados em criar uma árvore a partir do gráfico dado, tal que essa árvore conecte todos os vértices terminais pré-definidos. Mais precisamente , na variação que estamos trabalhando além dos vértices terminais nós adicionaremos mais duas restrições, uma delas é o número de Elos (que são os nós que conectam dois outros nós), e a outra o número de Roteadores (que são os nós que conectam mais de dois nós). A Figura 1 mostra um exemplo de uma árvore que conecta todos os nós terminais(marcados em preto) utilizando um roteador e dois elos.

Problema da Árvore de Steiner Limitada por Roteadores e Elos
INSTÂNCIA: Um grafo $G = (V, E)$ um conjunto de terminais T e dois inteiros R e L definindo os roteadores e elos
QUESTÃO: É possível conectar todos os terminais usando no máximo R roteadores e L elos?



(a) Instância.



(b) Solução.

Figura 1: Exemplo de solução.

2 Pré-Processamento

Nesta seção é proposto um pré-processamento que identifica subgrafos que com certeza não farão parte da solução, pois não conectam terminais de nenhuma forma ou existem outros nós que fazem isso melhor.

2.1 Remoção de folhas não terminais

A Figura 2 mostra umas sequencia de folhas sendo removidas, pois elas com certeza não farão parte da nossa resposta, pois não conectam nenhum nó terminal no caminho entre elas , são eles os nós: 8, 10, 11, 12, 13, e 16.
Essa remoção funciona da seguinte forma: Após executarmos um bfs para calcular o grau de cada Vértice, colocamos todos os vértices e seus respectivos graus em uma fila de prioridades, e enquanto o topo da fila tiver grau 1 e ele não for um nó terminal nós eliminaremos a aresta que conecta ele ao grafo, atualizando assim o grau do outro nó que faz parte dessa aresta, e inserindo esse nó na fila com seu novo grau.

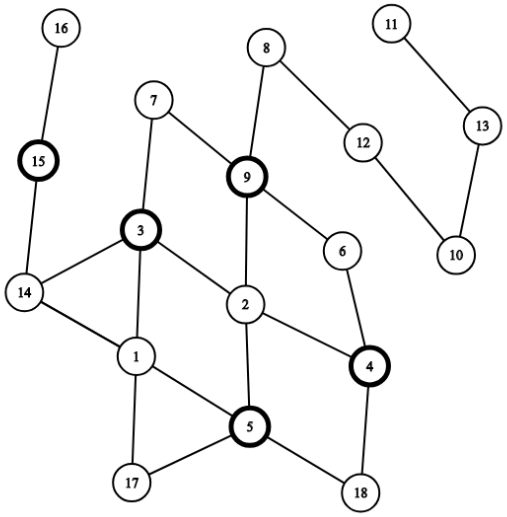
2.2 Remoção de nós de grau 2 que forma ciclos com seus vizinhos

A Figura 3 mostra o nó 17 removido, esse nó em particular é um nó de grau 2 que forma um ciclo com os nós 1 e 5, logo ele é irrelevante pois só serve de conexão para esses dois nós que já estão conectados diretamente.
Semelhante ao procedimento de remover os nós de grau 1 removeremos também os nós de grau 2 a partir da mesma fila de prioridade, quando os nós de grau 2 forem encontrados e não forem terminais verificaremos se os dois nós conectados a eles tem uma conexão direta, caso isso aconteça nós removeremos as duas arestas que incidem nesse nó, pois ele não fará parte da nossa resposta ótima.

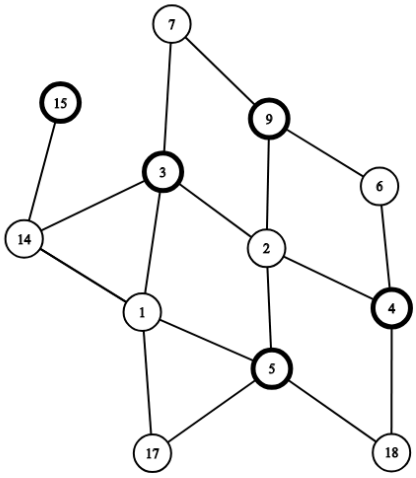
3 GRASP

Greedy Randomized Adaptive Search Procedure (GRASP) é uma meta-heurística em que cada iteração consiste de duas fases: uma para a construção de uma solução inicial e outra que busca melhores soluções a partir desta (Feo & Resende, 1995). A solução geral é atualizada sempre que a fase de busca encontrar uma solução melhor que todas as outras já encontradas. O processo é repetido até um certo número de iterações ou até um critério de parada estabelecido ser satisfeito. Nos últimos anos, o GRASP tem sido aplicado com sucesso em uma grande variedade de problemas de otimização (Resende & Ribeiro, 2005).

Neste trabalho, os métodos construtivos serão baseado nos algoritmos de Kruskal e Prim para Árvore Geradora Mínima. A entrada consiste num grafo sem pesos , uma sequencia de nós terminais e dois inteiros informando a quantidade de elos e roteadores. Esse grafo será dividido em 4 grupos que classificaremos da seguinte forma:
Grupo 1: Candidatos a roteadores(nós que conectam mais de 1 2 terminais)
Grupo 2: Candidatos a elos fortes (nós que conectam 2 terminais)
Grupo 3: candidatos a elos fracos (nós que conectam 1 terminal)
Grupo 4: outros.
A partir desses grupos aplicaremos uma seleção pseudo-aleatória junto ao método construtivo.

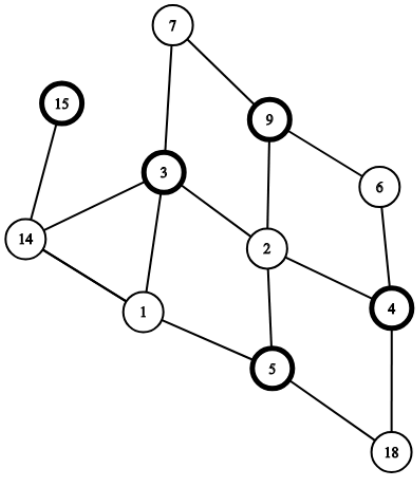


(a) Instância.



(b) Folhas Removidos.

Figura 2: Remoção de nós dispensáveis.



(a) Nós de grau 2 que fazem ciclos Removidos.

Figura 3: Remoção de nós dispensáveis.

3.1 Método Construtivo

Dado como entrada o grafo pré-processado G e os 4 grupos pré definidos, usaremos os métodos construtivos baseados nos algoritmos de Kruskal e Prim para construir a árvore geradora, a diferença principal é que a cada iteração escolheremos de forma aleatória um desses grupos (em que seus elementos foram bagunçados aleatoriamente entre si) e adicionaremos a aresta desse grupo na construção da árvore.

3.2 Busca Local

Após o método construtivo, inicia-se o método de Busca Local selecionando arestas a serem proibidas na proxima iteração, e a partir dai reconstruindo o pedaço ”destruido”da árvore, construindo assim seus vizinhos.
Essa construção depende do algoritmo proposto baseado no algoritmo de Kruskal ou no Prim. Para cada vizinho nós calculamos sua penalidade, e o menor entre os vizinhos e a instância inicial será atualizado como a melhor resposta.

4 Conclusão

Neste trabalho foram propostos duas meta-heurísticas semelhantes mas baseadas em algoritmos diferentes, além de um pré-processamento para otimização de tempo excluindo subgrafos irrelevantes.

Para trabalhos futuros serão adicionadas duas principais propostas e mudanças.
A primeira é a adição de uma nova técnica de pré-processamento , fixando algumas arestas obrigatórias, que são as pontes nos grafos que conectam subconjuntos que contém terminais.
A outra proposta é a adição de pesos nas arestas para atingir um problema mais complexo e mais comum do ponto de vista da literatura.

Referências

Feo, T. & Resende, M. (1995), 'Greedy randomized adaptive search procedures', *Journal of Global Optimization* **6**, 109–133.

Resende, M. & Ribeiro, C. (2005), *Metaheuristics: Progress as real problem solvers*, Springer, chapter GRASP whit path-relinking: recent advances and applications, pp. 29–63.

Pesquisa Operacional
Instituto de Computação
Universidade Federal de Alagoas

Histórico do Artigo:
Versão 1
Entregue: 9 de abril de 2019