Resumo de Artigo

A 2k-Vertex Kernel for Maximum Internal Spanning Tree

Ulysses Rocha - 191068
MO829B - Tópicos em Teoria da Computação - 2016.2

1 Introdução

Este resumo é feito sobre o artigo *A 2k-Vertex Kernel for Maximum Internal Spanning Tree* escrito por Wenjun Li, Jianxin Wang, Jianer Chen e Yixin Cao em 2014.

Os lemas, teoremas e definições do artigo original tiveram suas numerações preservadas.

No trabalho *A linear vertex kernel for maximum internal spanning tree*, Fomin et al. apresenta uma regra de redução que resulta em um kernel de $3k$-vértices, o lema 3.1 mostra que para esta redução é necessário encontrar em G um conjunto independente L' tal que $|L'| \geq 2|N_G(L')|$ o que é demonstrado trivial quando $|V(G)| \geq 3k - 3$.

O presente artigo então demonstra uma forma de obter algo ainda mais restrito da mesma regra de redução, baseando-se no uso de uma árvore geradora maximal, a qual permite encontrar uma estrutura redutível mesmo quando $2k < n < 3k - 3$. O Teorema 1.2 através dos lemas 3.5 e 3.1 afirmam que em um grafo com n-vértices, podemos encontrar uma árvore geradora com pelo menos $n/2$ vértices internos ou uma estrutura redutível. Assim temos que aplicando as regras de redução exaustivamente em uma arvore geradora maximal chegamos a uma solução ou a um grafo limitado a 2k-vértices. Uma particularidade que vale tomar nota é que esta kernelização nunca resulta na conclusão de que temos uma instância impossível para o problema.

Outro ponto importante que será apresentado no teorema 1.3 presente na sessão de conclusão deste resumo, é que fazendo uso do algoritmo de Nederlof com complexidade $O(2^n)$, poderemos resolver o *k-internal spanning tree problem* com complexidade $O(4^k n^{O(1)})$ devido a esta limitação do tamanho do kernel, sendo este o melhor resultado conhecido até então para o problema.

1.1 O Problema

O *k-internal spanning tree problem* pode ser definido como o problema de encontrar uma árvore geradora com pelo menos k nós internos em um grafo simples e não direcionado.

Note que decidindo a existência de uma árvore geradora com $n - 2$ vértices internos em um grafo com n vértices, conseguimos decidir a existência de um caminho hamiltoniano no grafo. Temos assim *k-internal spanning tree* um problema NP-Difícil.

1.2 Trabalhos Anteriores

Prieto e Sloper apresentaram uma primeira kernelização para o problema em 2003 de $O(k^3)$-vértices e em 2005 melhoram para $O(k^2)$-vértices. Em 2013 Fomin et al. apresenta um kernel de $3k$-vértices.

2 Pré-Processamento

Esta seção é o foco do artigo, apresentaremos as operações para chegar em uma árvore geradora maximal. Posteriormente essa árvore será utilizada pelo algoritmo para obter um $2k$-vértice kernel para o problema.

Aqui sumarizamos as principais definições utilizadas no artigo:

- Uma árvore T é uma arvore geradora de G se $V(T) = V(G)$ e $E(T) \subseteq E(G)$; arestas de G que não estão em T, como $E(G) \setminus E(T)$ são arestas da co-árvore (tradução livre de *cotree*) de T.
- Um vértice $u \in V(t)$ é considerado folha de T se $d_T(u) = 1$, caso contrário é um vértice interno.
- Um vertice interno u de T é um *branchpoint* se $d_T(u) \geq 3$.
- $L(T)$ e $I(T)$ denotam respetivamente os conjuntos de vértices folhas e vértices internos de T. $I_3(T)$ é o conjunto de branchpoints em T, $I_2(T)$ é o conjunto de vértices com grau 2. Temos então que $V(T)$ está particionado em $I_3(T)$, $I_2(T)$ e $L(T)$

2.1 Procedimento Guloso de Busca Local

Temos que $|I(T)| = |V(T)| - |L(T)|$, logo, maximizar o número de vértices internos é equivalente a minimizar o numero de folhas. Podemos fazer uso do seguinte fato nos auxilia a calcular o número de folhas de T:

$$|L(T)| - 2 = \sum_{v \in I(T)} (d_T(v) - 2) = \sum_{v \in I_3(T)} (d_T(v) - 2) \tag{1}$$

O artigo apresenta que para reduzir o número de folhas precisamos reduzir o número de *branchpoints* em T, aplicando trocas locais sob as folhas com o objetivo de aumentar o número de nós internos. A ação de trocas locais é definida como a substituição de uma aresta $E(T)$ por uma aresta da co-arvore de T. A ideia da troca é a de buscar entre as folhas de T uma ligação na co-árvore de T, e se houver, substituir um *branchpoints* no caminho entre essas folhas pela nova aresta da co-árvore, assim reduzindo o grau do vértice interno e consequentemente o número de folhas em T. Assim temos a primeira regra de troca:

Regra de Troca 1 (RT1) *Se existe uma aresta da co-arvore conectando duas folhas l_1 e l_2 de T, então busque uma aresta uv de $P_T(l_1, l_2)$ tal que o vértice u seja um branchpoint, e substitua $l_1 l_2$ por uv em T.*

Note que a menos que T seja uma árvore degenerada, haverá um *branchpoints*. Caso T não tenha *branchpoints* e seja somente um caminho entre as duas folhas, então já temos a árvore com o máximo de nós internos possível e o problema já está resolvido.

Outro ponto a se destacar é que após uma exaustiva aplicação desta redução, temos que todas as folhas restantes de T não são adjacentes em G. (Este fato vai auxiliar futuramente a encontrar um conjunto independente)

Definição 1 *Uma aresta da co-arvore de T é boa se ela conecta uma folha l a um vertice interno w de T. Nós dizemos assim que lw atravessa todas as arestas no caminho $P_T(l, w)$*

Definição 2 *Todos os vértices em $I_3(T)$ são separáveis. Um vértice $w \in I_2(T)$ é separável se existe um na co-árvore uma aresta boa lw de T que satisfaça pelo menos um dos seguintes:*

1. *$P_T(l, w)$ passa por um branchpoint*
2. *Algum vértice v em $P_T(l, w)$ é incidente na co-árvore a uma aresta boa $l'v$ de T onde $l' \neq l$*

Para os conjuntos separáveis, o artigo define $D(T)$ conforme o seguinte:

- Seja $D(T)$ o conjunto de todos os vértices separáveis em T, e $D_2(T)$ todos os vértices separáveis em $I_2(T)$. Logo $D(T) = D_2(T) \cup I_3(T) \subseteq I(T)$

Regra de Troca 2 (RT2) *Seja uv uma aresta atravessada por uma aresta boa lw da co-árvore de T. substitua lw por uv se satisfizer alguma das seguintes restrições:*

- $u \in I_3(T)$, e
  - $v = w$ ou $v \in I_3(T)$; ou
  - $v \in D_2(T)$, e existe uma aresta boa $l_v v$ na co-árvore, onde $l_v \neq l$.
- $u \in D_2(T)$, existe uma aresta boa $l_u u$ na co-árvore onde $l_u \neq l$.
  - $v = w$ ou $v \in I_3(T)$; ou
  - $v \in D_2(T)$ e existe uma aresta boa $l_v v$ na co-árvore onde $l_v \notin \{l, l_u\}$

Após a aplicação da regra de troca 2, deve-se voltar a tentar aplicar a regra 1.

A árvore geradora T é chamada de *maximal* quando não houverem mais operações de troca possíveis.

Para demonstrar que é possível chegar em uma árvore geradora maximal em tempo polinomial, é apresentado o lema 2.1 e o teorema 2.2.

Lema 2.1 *Aplicando RT1 ou RT2 a uma árvore geradora T de G resulta em uma nova árvore geradora T' onde $|I(T')| \geq |I(T)| + 1$*

Comentando brevemente a prova, temos que nenhuma das reduções isolam vértices da árvore, assim, ambas preservam as características de T' como árvore geradora de G. Em RT1 ao menos uma das folhas se tornará nó interno da árvore, diminuindo ao menos uma folha e assim preservando a afirmação do lema. Com RT2 temos que ou haverá um aumento no número de vértices internos ou teremos uma situação onde RT1 poderá ser novamente aplicada na árvore geradora.

Sabendo que cada redução é feita em tempo polinomial, como temos que o número de vértices internos de um grafo G é limitado por $|V(G)| - 2$ e que cada aplicação do teorema aumenta o número de vértices internos, concluímos que toda a regra de redução que gera a árvore maximal é feita em tempo polinomial.

Teorema 2.2 *A árvore geradora maximal pode ser construida em tempo polinomial*

3 Kernelização

Essa sessão mostra as regras de redução, e as implicações do uso delas em uma árvore geradora maximal, assim, teremos a conclusão de que será possível para uma estrutura de até $2k$-vértices separando a árvore geradora maximal em sub-árvores e aplicando a cada componente as reduções.

3.1 Regras de kernelização do artigo de Fomin et al.

A regra de redução apresentadas na kernelização é a mesmas definida no artigo *"A linear vertex kernel for maximum internal spanning tree"* por Fomin et al.

Seja $opt(G)$ o número máximo de nós internos que uma árvore geradora de G, temos:

Lema 3.1 *Seja L' um conjunto independente de G, tal que $|L'| \geq 2|N_G(L')|$. podemos encontrar em tempo polinomial conjuntos não vazios $S \subseteq N_G(L')$ e $L \subseteq L'$, tal que :*

1. *$N_G(L) = S$, e*
2. *O grafo $(S \cup L, E(G) \cap (S \times L)))$ possui uma árvore geradora onde todos os vértices de S e $|S| - 1$ vértices de L são internos.*

Além disso, seja G' obtido de G adicionando v_s adjacente a cada vértice em $N_G(S) \setminus L$, adicionando o vértice v_l adjacente a v_s, e removendo todos os vértices de $S \cup L$, temos $opt(G') = opt(G) - 2|S| + 2$.

Pelo lema 3.1, temos a garantia da segurança da regra de redução a seguir:

Regra de Redução *Encontre um subconjunto não vazio S e L de vértices como no lema 3.1. Retorne (G', k') onde G' é definido no lema 3.1, e $k' = k - 2|S| + 2$*

3.2 Implicações da árvore geradora maximal na kernelização

O artigo expõe que o grande obstáculo da kernelização é identificar o conjunto L', tal que $|L'| \geq 2|N_G(L')|$, essa tarefa é trivial quando $|V(G)| \geq 3k - 3$, mas é bastante difícil quando $2k < |V(G)| < 3k - 3$. A abordagem adotada é a de separar a árvore geradora maximal T em várias sub-árvores e calcular $L(T)$ pelo número de $I(T)$ em cada sub-árvore individualmente.

Os lemas e corolários da sessão de kernelização do artigo original, concluem no seguinte lema, que usaremos para fazer as conclusões.

Lema 3.5 *Seja T uma árvore geradora maximal de G onde $|V(G)| \geq 4$. Se $|L(T)| > |I(T)|$, então podemos achar em tempo polinomial um conjunto independente L' de G tal que $|L'| \geq 2|N_G(L')|$.*

4 Conclusões

Através da regra de redução, conseguiremos aplicando-a exaustivamente limitar o tamanho da instância até um $n/2$ vértices internos. Limitando n a k concluindo então a principal afirmação do artigo, a existência de um kernel de 2k-vértices.

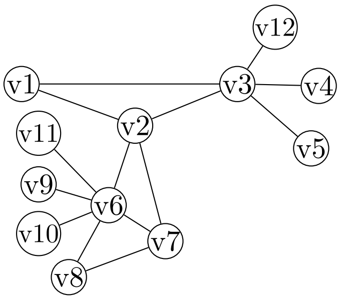
Teorema 1.2 *Dado um grafo G com n-vértices, podemos encontrar em tempo polinomial ou uma árvore geradora de G com pelo menos $n/2$-vértices internos ou uma estrutura redutível.*

Teorema 1.1 *O problema da k-internal spanning tree tem um kernel de 2k-vértices*

Após a obtenção desse kernel, utilizando o algoritmo de Nederlof, com complexidade $O(2^n n^{O(1)})$, teremos n limitado a 2k, assim, $O(2^{2k} n^{O(1)})$, resultando em $O(4^k n^{O(1)})$.

Teorema 1.3 *O problema k-internal spanning tree pode ser resolvido em tempo $O(4^k n^{O(1)})$*

5 Exercício



Questão Aplique no grafo apresentado as regras de troca 1 e 2, justificando as operações para obter uma árvore geradora maximal. Aplique a regra de kernelização uma vez e mostre o grafo resultante.