A Linear Time Algorithm for the Minimum Spanning Caterpillar Problem for Bounded Treewidth Graphs

Michael J. Dinneen Masoud Khosravani

(SIROCCO 2010)

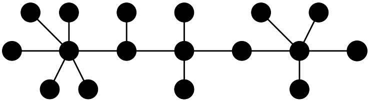
Department of Computer Science
University of Auckland

Outline

Caterpillars

Definition

By a caterpillar we mean a tree that reduces to a path by deleting all its leaves. We refer to the remaining path as the *spine* of the caterpillar. The edges of a caterpillar H can be partitioned to two sets, the spine edges, $\mathcal{B}(H)$, and the leaf edges, $\mathcal{L}(H)$.



Minimum Spanning Caterpillar Problem

Definition (Cost Function)

Let $G = (V, E)$ be a graph. Also let $b : E \rightarrow \mathbb{N}$ and $l : E \rightarrow \mathbb{N}$ be two (cost) functions. For each caterpillar H as a subgraph of G we define the cost of H by

$$c(H) := \sum_{e \in \mathcal{B}(H)} b(e) + \sum_{e' \in \mathcal{L}(H)} l(e').$$

Problem

Minimum Spanning Caterpillar Problem (MSCP): find a caterpillar with the minimum cost that contains all vertices.

Applications and Complexity

- Finding a cost effective subnetwork within a large network.
- Tan and Zhang used it to solve some problems concerning the Consecutive Ones Property problem.

Computational Complexity:

- NP-hard for general graphs (by a straightforward transformation from the Hamiltonian Path Problem).

Reference

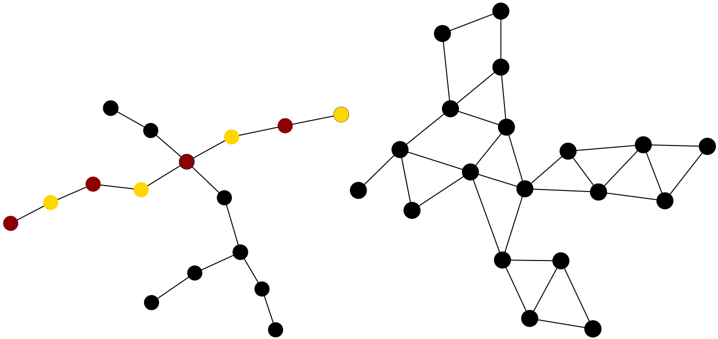
J. Tan and L. Zhang, The Consecutive Ones Submatrix Problem for Sparse Matrices, Algorithmica, 48, 3, 2007, 287-299.

Motivation for Tree and Path Decomposition

- Many hard problems on general graphs has polynomial solution for trees.
  - Example: Vertex Colouring.
- What does it mean that a graph looks like a tree?

Motivation for Tree and Path Decomposition

- Many hard problems on general graphs has polynomial solution for trees.
  - Example: Vertex Colouring.
- What does it mean that a graph looks like a tree?



Tree and Path Decomposition

Definition

Let $G = (V, E)$ be graph, a tree decomposition of G is a pair (X, T), where $X = \{X_1, ..., X_n\}$ is a family of subsets of V, and T is a tree (path) whose nodes are the subsets X_i, satisfying the following properties:

1. The union of all sets X_i equals V. That is, each graph vertex is associated with at least one tree node.
2. For every edge (v, w) in the graph, there is a subset X_i that contains both v and w.
3. If X_i and X_j both contain a vertex v, then all nodes X_z of the tree in the (unique) path between X_i and X_j contain v.

Reference

N. Robertson and P. Seymour, Graph minors. II. Algorithmic aspects of tree-width, Journal of Algorithms, 7, 1986.

Treewidth and Examples

Definition

Let G be graph. The width of a tree (path) decomposition (T, X) of G is $max\{|X_i| - 1 \mid X_i \in X\}$. The treewidth $tw(G)$ (pathwidth $pw(G)$) of G is the minimum width among all possible tree (path) decompositions of G.

Examples

1. The treewidth of each tree is one. But pathwidth is not a fixed number.
2. The treewidth of each outer-planar graph is two.
3. The treewidth (and pathwidth) of K_n is $n - 1$.

k-parse

Definition

- A *k+1-boundaried graph* is a pair (G, ∂) of a graph $G = (V, E)$ and an injective function ∂ from $\{0, \dots, k\}$ to V. The image of ∂ is the set of *boundaried vertices* and is denoted by $Im(\partial)$.

Given a path decomposition of width k of a graph, one can represent the graph by using strings of (unary) operators from the following *operator set* $\Sigma_k = V_k \cup E_k$:

$$V_k = \{ \textcircled{0}, \dots, \textcircled{k} \} \quad \text{and} \quad E_k = \{ \boxed{i\ j} \mid 0 \le i < j \le k \}.$$

Where V_k is the set of vertex operators and E_k is the set of edge operators.

k-parse(continued)

To generate a graph from a smooth tree decomposition of width k, an additional (binary) operator $\oplus$, called *boundary join*, is added to Σ_k. The semantics of these operators on $(k+1)$-boundaried graphs G and H of are as follows:

- G ⓘ — Add an isolated vertex to the graph G, and label it as the new boundary vertex i.
- G $\boxed{i\ j}$ — Add an edge between boundaried vertices i and j of G (ignore if operation causes a multi-edge).
- $G \oplus H$ — Take the disjoint union of G and H except that equal-labeled boundary vertices of G and H are identified.

Reference

M. J. Dinneen, Practical enumeration methods for graphs of bounded pathwidth and treewidth, Center for Discrete Mathematics and Theoretical Computer Science, CDMTCS–055,1997,Auckland

Algorithm

1. We use a forest of at most $k+1$ different caterpillars as a partial solution.
2. Each partial solution has at least one vertex in the boundary set $\partial = \{0, \ldots, k\}$.
3. We code the information of each partial solution in a state vector $S = (A, B)$. Where A is a $(k+1)$-tuple $(a_0, \ldots, a_k)$. Each a_i represents a label for the boundary vertex i from the set $\{H, S, C, I, L\}$.
4. The labels H, S, C, I, and L are characteristics of the boundary vertices in a partial solution. They stand for *head, spine, center* (of a star), *isolated vertex*, and *leaf*, respectively.
5. The set B is a partition set of ∂. If any two boundary vertices belong to the same element of B, then they belong to the same connected component of a partial solution that is represented by B.

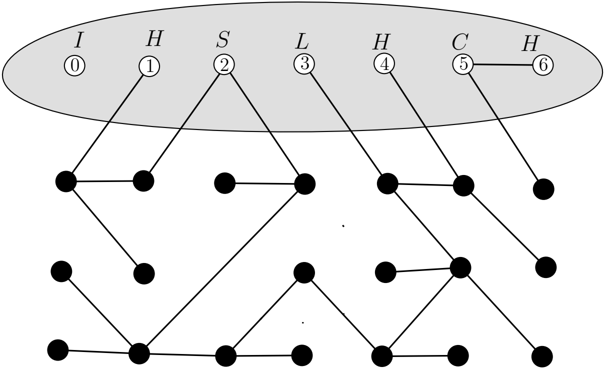


Figure: A forest of caterpillars as a partial solution,
$A = (I, H, S, L, H, C, H)$ and $B = \{\{0\}, \{1, 2, 3, 4\}, \{5, 6\}\}$.

Algorithm

Now we follow a dynamic programming approach by keeping information in a table with respect to the following operations (where rows indexed by states and columns indexed by operations):

1. Introducing a new vertex by a vertex operation.
2. Introducing a new edge by edge operation.
3. Introducing branches by boundary join operation.

Rules for vertex operation

We filter representatives (A, B) depending on whether the vertex operator (i) disconnects the currently kept partial spanning tree.

- If $B_i - \{i\}$ is empty or if it contains boundary vertices that are labeled just as leaves, then the partial solution becomes disconnected. [filter case]
- Otherwise, we update the representative entry (A', B') with $B' = (B - \{B_i\}) \cup \{\{B_i - \{i\}\}, \{i\}\}$ and A' is the same as A except it has I in its i-th coordinate. [update case]

Table 1: Rules for edge operation

Rule	a_i	a_j	a'_i	a'_j	new best cost
1	H	H	S	S	$\min\{x + b(\{i,j\}), x'\}$
2(a)	H	C	S	H	$\min\{x + b(\{i,j\}), x'\}$
2(b)	H	C	S	S	$\min\{x + b(\{i,j\}), x'\}$
3(a)	H	I	H	L	$\min\{x + l(\{i,j\}), x'\}$
3(b)	H	I	S	H	$\min\{x + b(\{i,j\}), x'\}$
4(a)	C	C	H	H	$\min\{x + b(\{i,j\}), x'\}$
4(b)	C	C	H	S	$\min\{x + b(\{i,j\}), x'\}$
4(c)	C	C	S	S	$\min\{x + b(\{i,j\}), x'\}$
5(a)	C	I	S	H	$\min\{x + b(\{i,j\}), x'\}$
5(b)	C	I	C	L	$\min\{x + l(\{i,j\}), x'\}$
6	S	I	S	L	$\min\{x + l(\{i,j\}), x'\}$
7	I	I	C	H	$\min\{x + b(\{i,j\}), x'\}$

Table 2: Rules for boundary join operation

Rule	a_i	a'_i	a''_i
1	S	$\{H, I, C\}$	S
2	H	$\{I, C\}$	H
3	H	H	S
4	C	$\{C, I\}$	C

Correctness and Complexity

Theorem

The algorithm solves the spanning caterpillar problem in time $O(5^{k+1}B_{k+1}n)$ for a graph of bounded pathwidth k and in $O(5^{2k+2}B_{k+1}^2 n)$ for a graph of bounded treewidth k; where n is the number of vertices and B_{k+1} is the $(k+1)$th Bell number (the number of partitions of a set with $k+1$ members).

Lemma (1)

Let $G = (g_0, \ldots, g_m)$ be a graph whose $tw(G) = k$ and also let $\mathcal{T}$ be the table produced by the algorithm. If $\mathcal{T}((A, B), g_m)$ has a true entry in the last column of $\mathcal{T}$ such that $B = \partial$, then the graph G has a spanning caterpillar.

Lemma (2)

Let $G = (g_0, \ldots, g_m)$ be a graph whose $pw(G) = k$. If G that has a spanning caterpillar then the last column of the table $\mathcal{T}$ that results from the algorithm has a true entry $\mathcal{T}((A, B), g_m)$ with $B = \partial$.

Lemma (3)

Let $G = H \oplus H'$, where H and H' are graphs with treewidths at most k. If G has a spanning caterpillar then the column of the table $\mathcal{T}$, that results from applying the algorithm to G, has a true entry $\mathcal{T}((A, B), H \oplus H')$ with $B = \partial$.

Related Problems

- Minimum Spanning Ring Star Problem: the goal is to find a minimum spanning subgraph (star ring) that consists of a cycle and vertices of degree one that are connected to it.
- Dual Cost Minimum Spanning tree: where the cost of an edge incident to a leaf is different from the other edges.

References

Roberto Baldacci, Mauro Dell'Amico and J. Salazar González, The Capacitated -Ring-Star Problem, Operations Research, 55, 6 ,2007.

-

M. Labbé, G. Laporte, I. Rodríguez Martín and J. J. Salazar González, The Ring Star Problem: Polyhedral analysis and exact algorithm, Networks, 43, 3, 2004.

Conclusion

- We presented an algorithm that efficiently finds a minimum spanning caterpillar in some classes of graphs that have small treewidth, like outerplanar, series-parallel and Halin graphs.
- Our algorithm can be easily modified to solve other related network problems like the Minimum Ring Star Problem and the Dual Cost Minimum Spanning Tree.
- Question: How we can improve the time complexity by reducing the size of a state table?