Optimal Algorithms for the Path Cover Problem on P_4-sparse Graphs

Katerina Asdre Stavros D. Nikolopoulos Charis Papadopoulos

Department of Computer Science, University of Ioannina
P.O.Box 1186, GR-45110 Ioannina, Greece

{asdre, stavros, charis}@cs.uoi.gr

Abstract

In this paper we present optimal algorithms for finding the smallest number of vertex-disjoint paths that cover the vertices of a P_4-sparse graph. Specifically, given the modular decomposition tree of a P_4-sparse graph G on n vertices and m edges, we first present an $O(n)$-time simple sequential algorithm for the path cover problem and, then, using standard tree contraction and bracket matching techniques [9], we describe an optimal parallel algorithm which runs in $O(\log n)$ time with $O(n/\log n)$ processor on the EREW PRAM mode. Our results generalize previous results [9], and extend the family of perfect graphs admitting optimal solutions for the path cover problem.

Key words: P_4-sparse graphs, modular decomposition, path cover, algorithms.

1 Theoretical Framework

The *modular decomposition tree* $T(G)$ of a graph G (or *md-tree* for short) is a unique labeled tree associated with the modular decomposition of G in which the leaves of $T(G)$ are the vertices of G and the set of leaves associated with the subtree rooted at an internal node induces a strong module of G. Thus, the md-tree $T(G)$ represents all the strong modules of G. An internal node is labeled by either P (for *parallel* module), S (for *series* module), or N (for *neighborhood* module). It is shown that for every graph G the md-tree $T(G)$ is unique up to isomorphism, it has $O(V(G))$ nodes and it can be constructed sequentially in linear time [1,8].

Let t be an internal node of the md-tree $T(G)$ of a graph G. We denote by $M(t)$ the module corresponding to t which consists of the set of vertices of G associated with the subtree of $T(G)$ rooted at node t; note that $M(t)$ is a

Preprint submitted to CTW 2005 *25 March 2005*

strong module for every (internal or leaf) node t of $T(G)$. Let $u_1, u_2, \ldots, u_p$ be the children of the node t of $T(G)$. We denote by $G(t)$ the *representative graph* of the module $M(t)$ defined as follows: $V(G(t)) = \{u_1, u_2, \ldots, u_p\}$ and $u_i u_j \in E(G(t))$ if there exists an edge $v_k v_\ell \in E(G)$ such that $v_k \in M(u_i)$ and $v_\ell \in M(u_j)$. $G(t)$ is an edgeless graph if t is a P-node, $G(t)$ is a complete graph if t is an S-node, and $G(t)$ is a prime graph if t is an N-node (see also [2]).

A graph G is called a *spider* if its vertex set $V(G)$ admits a partition into sets S, K, and R; the triple (S, K, R) is called *spider-partition* [2,5]. A graph G is a *prime spider* if G is a spider with $|R| \leq 1$. If the condition of case P3(i) holds then the spider G is called *thin spider*, whereas if the condition of case P3(ii) holds then G is called *thick spider*; note that the complement of a thin spider is a thick spider and vice versa. For the class of P_4-sparse graphs [3], Giakoumakis and Vanherpe [2] showed the following result:

Lemma 1.2. (Giakoumakis and Vanherpe [2]): *Let G be a graph and let $T(G)$ be its modular decomposition tree. The graph G is P_4-sparse iff for every N-node t of $T(G)$, $G(t)$ is a prime spider with a spider-partition (S, K, R) and no vertex of $S \cup K$ is an internal node in $T(G)$.*

Based on the structural properties of a prime spider, it is not difficult to see that the following result holds:

Lemma 1.3. *Let G be a P_4-sparse graph on n vertices and let $T(G)$ be its modular decomposition tree. The spider partitions (S_i, K_i, R_i) of all the internal N-nodes t_i of $T(G)$ can be found in $O(n)$ time.*

Let $T(G)$ be the md-tree of a P_4-sparse graph G. We compute in each N-node $t_i \in T(G)$, the sets S_i, K_i, R_i of $G(t_i)$, we check if $G(t_i)$ is a thin or a thick spider and we set appropriate labels in each N-node $t_i \in T(G)$; we denote $T^*(G)$ the resulting md-tree. Based on the techniques described in [7,9], we modify further the md-tree $T^*(G)$: we binarize the tree $T^*(G)$ in such a way that each of its internal nodes labelled by either P or S has exactly two children; we denote $T^*_b(G)$ the resulting tree. The left and right child of an internal P-node or S-node u of $T^*_b(G)$ will be denoted by v and w, respectively.

Let $G[M(u)]$ denote the subgraph induced by the leaf descendants of u in $T^*_b(G)$, and let $L(u)$ denote the number of leaf descendants of u in $T^*_b(G)$, that is, the number of vertices in $G[M(u)]$. We say that $T^*_b(G)$ is *leftist*, denoted by $T^*_{bl}(G)$, if for every internal node u labelled by either P or S, the condition $L(v) \geq L(w)$ is satisfied. For every S-node u of $T^*_{bl}(G)$, we replace the subtree rooted at node w (the right child of u) with the $L(w)$ leaves and call the resulting tree the *reduced* leftist binary tree of $T^*_{bl}(G)$; we denote it by $T^*_{blr}(G)$.

Let $\lambda(u)$ denote the number of paths in the minimum path cover of the graph $G[M(u)]$. It is easy to see that, in order to construct the path cover using the tree $T^*_{blr}(G)$, we need to know the number of paths $\lambda(u)$ of each internal node $u \in T^*_{blr}(G)$. Recall that, if u is a P-node or S-node then it has a left child v and a right child w; otherwise, u is an N-node and it has at least 4 children which induce a prime spider $G(u) = (S, K, R)$ with either $R = \emptyset$ or $R = \{r\}$. Based on the results of [7] and [4], we obtain the following formula for the number of paths of a P_4-sparse graph.

$$\lambda(u) = \begin{cases} \lambda(v) + \lambda(w) & \text{if } u \text{ is a P-node,} \\ \max\{1,\ \lambda(v) - L(w)\} & \text{if } u \text{ is an S-node,} \\ \lambda(r) + \left\lceil \max\left\{0,\ \frac{|K|-2\lambda(r)}{2}\right\}\right\rceil & \text{if } G(u) \text{ is a thin spider,} \\ \max\{1,\ \lambda(r)\} & \text{if } G(u) \text{ is a thick spider.} \end{cases} \tag{1}$$

Lemma 1.5. *Let G be a P_4-sparse graph on n vertices and let $T(G)$ be its modular decomposition tree. The reduced leftist binary tree $T^*_{blr}(G)$ and the number of paths in a minimum path cover of G can be computed in $O(n)$ time.*

2 Minimum path cover in P_4-sparse graphs

In this section we review some ideas for finding a minimum path cover of a P_4-sparse graph G. We suppose that the reduced leftist binarized tree $T^*_{blr}(G)$ of the graph G is given. We focus on the internal N-nodes since the cases of the P-nodes and S-nodes have already been established [6].

Let u be an internal N-node of $T^*_{blr}(G)$. Let $\mathcal{P}$ be the minimum path cover of the graph $G[M(u)]$ and let $\lambda(u)$ be the number of paths in $\mathcal{P}$, that is, $\lambda(u) = |\mathcal{P}|$; recall that, $M(u)$ is the module which corresponds to u and consists of all the vertices of G associated with the subtree of $T(G)$ rooted at node u.

Let a prime spider $G(u) = (S, K, R)$ with $S = \{s_1, s_2, \ldots, s_\ell\}$ and $K = \{k_1, k_2, \ldots, k_\ell\}$, where $|S| = |K| = \ell$. If R is empty, in Eq. (1) we set $\lambda(r) = 0$; otherwise, $R = \{r\}$ and $d = \lambda(r)$, and let $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_d\}$ be a minimum path cover of $G[M(r)]$. For every i, $1 \le i \le d$, let q_i and q_i' be the endpoints of Q_i. Then, for the computation of the minimum path cover $\mathcal{P}$ of $G[M(u)]$ we distinguish the following two cases.

Case 1: $G(u) = (S, K, R)$ is a thin spider. The prime spider $G(u)$ has $2\ell + 1$ vertices and there exists a bijection f such that $f(s_i) = k_i, 1 \le i \le \ell$. By Eq. (1), we have that the number of paths in a minimum path cover of $G[M(u)]$ are $\lambda(r) + \left\lceil \max\left\{0, \frac{\ell - 2\lambda(r)}{2}\right\}\right\rceil$. If R is empty, then the graph $G[S \cup K]$ contains $t = \lceil \frac{\ell}{2} \rceil$ paths and the minimum path cover of $G[M(u)]$ is:

$$\mathcal{P} = \{[s_1 k_1 k_2 s_2], [s_3 k_3 k_4 s_4], \ldots, [s_{\ell-1} k_{\ell-1} k_\ell s_\ell]\}. \tag{2}$$

If $R = \{r\}$, then the paths of $G[M(u)]$ are obtained by joining the end-points of some paths Q_i of $G[M(r)]$ with the vertices of the t paths of $G[S \cup K]$. Thus, if $t \leq d$ we have:

$$\mathcal{P} = \{[s_1 k_1 q_1 \dots q'_1 k_2 s_2], \dots, [s_{\ell-1} k_{\ell-1} q_t \dots q'_t k_\ell s_\ell], Q_{t+1}, Q_{t+2}, \dots, Q_d\}. \quad (3)$$

Otherwise, if $t > d$ then the paths that occur in a minimum path cover of $G[M(u)]$ are:

$$\mathcal{P} = \{[s_1 k_1 q_1 \dots q'_1 k_2 s_2], \dots, [s_{2d-1} k_{2d-1} q_d ... q'_d k_{2d} s_{2d}], \dots, [s_{\ell-1} k_{\ell-1} k_\ell s_\ell]\}. \quad (4)$$

Case 2: $G(u) = (S, K, R)$ is a thick spider. Let $t = \lceil \frac{\ell}{2} \rceil$. If R is empty, then the graph $G[S \cup K]$ is a hamiltonian graph and every edge in a hamiltonian path has one end-vertex in S and the other in K (i.e., there exists no hamiltonian path which contains an edge with both end-vertices in K). Thus, if t is an odd number we have,

$$\mathcal{P} = \{[s_1 k_\ell s_2 k_{\ell-1} \dots s_t k_{t-1} s_{t+1} k_t s_{t+2} k_{t-2} \dots s_{\ell-1} k_2 s_\ell k_1]\}, \quad (5)$$

otherwise,

$$\mathcal{P} = \{[s_1 k_\ell s_2 k_{\ell-1} \dots s_t k_{t+1} s_{t+2} k_t s_{t+1} k_{t-1} \dots s_{\ell-1} k_2 s_\ell k_1]\}. \quad (6)$$

If $R = \{r\}$, then the hamiltonian path of $G[S \cup K]$ is connected to the path Q_1 of $G[M(r)]$. Thus, the paths that occur in a path cover $\mathcal{P}$ of $G[M(u)]$ are:

$$\mathcal{P} = \{[s_1 k_\ell s_2 k_{\ell-1} \dots s_{\ell-1} k_2 s_\ell k_1 q_1 \dots q'_1], Q_2, Q_3, \dots, Q_d\}. \quad (7)$$

In both cases, the paths in $\mathcal{P}$ form a minimum path cover of $G[M(u)]$. Thus, given the paths $\mathcal{Q}$ of $G[M(r)]$, it is easy to describe a function that computes a minimum path cover in an N-node u of $T^*_{blr}(G)$; we call such a function *Path_Spider*$(u, \mathcal{Q})$. We store the paths of $\mathcal{P}$ in a doubly linked list which contains pointers to the first and last element of each path. Then, using basic list operations and based on the Eqs. (2)–(7), we can merge the minimum path cover $\mathcal{Q}$ of $G[M(r)]$ with a minimum path cover of the graph $G[S \cup K]$.

3 A simple optimal algorithm

Let G be a P_4-sparse graph on n vertices and m edges and let $T(G)$ be its md-tree. We note that the tree $T(G)$ can be computed in linear time, i.e., in $O(n + m)$ time, by using one of the well-known algorithms of [1,8]. The following algorithm is based on the function Path_Spider which is applied on an N-node. For a P-node or an S-node the algorithm uses appropriate functions proposed in [6]. Our path cover algorithm is the following:

Algorithm **Minimum_Path_Cover**

1. Compute the md-tree $T(G)$ of G and, then, compute the tree $T^*_{blr}(G)$;
2. For each internal node u of $T^*_{blr}(G)$, compute recursively the minimum path cover of the graph $G(u)$ as follows:
   if u is a P-node or an S-node **then**
   use the function described in [6] and compute a minimum path cover of $G(u)$;
   else { u is a N-node }
   use the function Path_Spider$(u, \mathcal{Q})$ and merge the paths of the minimum path cover $\mathcal{Q}$ of $G[R]$ with the paths of the minimum path cover of $G[S \cup K]$;

Theorem 3.1. *Let G be a P_4-sparse graph on n vertices and m edges. The minimum path cover of G can be computed in $O(n + m)$ time.*

4 An optimal parallel algorithm

Although the sequential algorithm is quite simple, a naive parallelization of this algorithm needs time proportional to the height of the md-tree $T(G)$, which is $O(n)$. In order to obtain an efficient parallel algorithm, we make use of the path tree structures and bracket matching technique introduced in [9].

A path tree is a rooted binary tree whose nodes are exactly the vertices of a path P in G. The path P can be efficiently obtained by the path tree using the inorder traversal. Thus, the corresponding path of a path tree can be constructed in parallel by applying the Euler tour technique.

In order to construct the path trees efficiently in a parallel process environment, we generate a sequence of square/round brackets for each node of $T^*_{blr}(G)$. The path trees are constructed by finding matching pairs of square brackets and matching pairs of round brackets independently. Note that, given a bracket sequence corresponding to the vertices of a graph G, the path trees and consequently the path cover of G can be constructed efficiently.

Let G be a P_4-sparse graph on n vertices, and let $T^*_{blr}(G)$ be the reduced leftist binary tree of $T(G)$; the tree $T^*_{blr}(G)$ can be constructed in $O(\log n)$ time using $O(n/\log n)$ processors on the EREW PRAM model. We next describe the bracket sequence assigned to an N-node u of $T^*_{blr}(G)$; the bracket assignment corresponding to a P-node or to an S-node has been described in [9]. Suppose that $G(u) = (S, K, R)$ is a thin spider. If $R = \emptyset$, then the path trees of $G[S \cup K]$ are constructed from a bracket sequence verifying Eq. (2). In the case where $R = \{r\}$, the graph $G[M(r)]$ contains $\lambda(r)$ paths, which are merged with the path trees of $G[S \cup K]$ as described in Eq. (3) and Eq. (4). Suppose now that $G(u) = (S, K, R)$ is a thick spider. Then, the hamiltonian path of $G[S \cup K]$ described by Eq. (5) and Eq. (6) is constructed from a specific path tree, rooted at a vertex of K, say k_1; each internal node of this path tree has only a left child. According to Eq. (7), we merge the path trees of the graph $G[M(r)]$ with the path tree of the graph $G[S \cup K]$; to this end, we set the root of the path tree of the path Q_1 of $G[M(r)]$ to be the right child of k_1. Concluding, in a parallel environment we have the following result.

Theorem 4.1. *Let G be a P_4-sparse graph on n vertices and let $T(G)$ be its modular decomposition tree. The minimum path cover of G can be computed in $O(\log n)$ time using $O(n/\log n)$ processors on the EREW PRAM model.*

References

[1] E. Dalhaus, J. Gustedt and R.M. McConnell, Efficient and practical algorithms for sequential modular decomposition, *J. Algorithms* **41** (2001) 360–387.

[2] V. Giakoumakis and J-M. Vanherpe, On extended P_4-reducible and P_4-sparse graphs, *Theoretical Comp. Science* **180** (1997) 269–286.

[3] C. Hoáng, Perfect graphs, Ph.D.thesis, McGill University, Montreal, Canada, 1985.

[4] W. Hochstätler and G. Tinhofer, Hammiltonicity in graphs with few P_4's , *Computing* **54** (1995) 213–225.

[5] B. Jamison and S. Olariu, Linear-time optimization algorithms for P_4-sparse graphs, *Discrete Appl. Math.* **61** (1995) 155–175.

[6] R. Lin, S. Olariu and G. Pruesse, An optimal path cover algorithm for cographs, *Comput. Math. Appl.* **30** (1995) 75–83.

[7] R. Lin, S. Olariu, J.L. Schwing and J. Zhang, A fast EREW algorithm for minimum path cover and hamiltonicity for cographs, *Parallel Algorithms Appl.* **2** (1994) 99–113.

[8] R.M. McConnell and J. Spinrad, Modular decomposition and transitive orientation, *Discrete Math.* **201** (1999) 189–241.

[9] K. Nakano, S. Olariu and A.Y. Zomaya, A time-optimal solution for the path cover problem on cographs, *Theoretical Comp. Science* **290** (2003) 1541–1556.