On the k-path cover problem for cacti [*]

Zemin Jin and Xueliang Li
Center for Combinatorics and LPMC
Nankai University
Tianjin 300071, P.R. China
zeminjin@eyou.com, x.li@eyou.com

Abstract

In this paper we investigate the k-path cover problem for graphs, which is to find the minimum number of vertex disjoint k-paths that cover all the vertices of a graph. The k-path cover problem for general graphs is NP-complete. Though notable applications of this problem to database design, network, VLSI design, ring protocols, and code optimization, efficient algorithms are known for only few special classes of graphs. In order to solve this problem for cacti, i.e., graphs where no edge lies on more than one cycle, we introduce the so-called Steiner version of the k-path cover problem, and develop an efficient algorithm for the Steiner k-path cover problem for cacti, which finds an optimal k-path cover for a given cactus in polynomial time.

Keywords: k-path cover; Steiner cover; efficient algorithm.

AMS subject classification (2000): 68R10, 05C70, 05C85, 90C27, 68Q17, 94C15.

1 Introduction

Let $G = (V, E)$ be an undirected simple graph. A *path cover* of G is a set of vertex disjoint paths which together cover all the vertices of G. The *path cover problem* is to determine a path cover of G that uses the minimum number of paths, denoted by $\pi(G)$. Naturally, $\pi(G) = 1$ if and only if G contains a Hamiltonian path, and so the path cover problem is NP-complete in general, and even for planar graphs, bipartite graphs, chordal graphs, and cubic graphs [6, 7]. On the other hand, there are many known special classes of graphs for which the path cover problem can be solved in polynomial time [1, 2, 5, 8, 9, 10, 12, 16]. The path cover problem finds applications to database design, network, VLSI design, ring protocols, code optimization, and mapping parallel programs to parallel architectures [3, 11, 15], etc.

For a given positive integer k, a path cover is called a *k-path cover* if each path in the path cover contains at most k vertices. The k-path cover problem

[*]Research supported by NSFC.

is to determine a k-path cover of a graph G that uses the minimum number of paths, denoted by $\pi_k(G)$. The k-path cover problem has applications in broadcasting, computer and communications networks [17], and vehicle routing problems [14]. It is easy to see that the 2-path cover problem is equivalent to the problem of finding a maximum matching of G, which can be solved in polynomial time. However, the problem is NP-complete in general for $k = 3$. As usually done for such problems, researchers focus on the problem on special classes of graphs for which the problem can be solved in polynomial time. However, there are only a few such classes of graphs known. Yan et al [17] gave a linear time algorithm for trees. Steiner [13] showed that the k-path cover problem is NP-complete for cographs if k is a part of the input, but it can be solved in polynomial time if k is fixed. Steiner [14] showed the NP-completeness of the problem for comparability graphs, and presented a polynomial algorithm for bipartite permutation graphs.

A graph is called a *cactus* if no edge lies on more than one cycle. Note that the class of cacti properly contains the class of trees. In this paper we focus on the k-path cover problem for cacti. Moran [10] developed an efficient algorithm for the optimal path cover of cacti. Though there is a linear time algorithm of Yan et al [17], it only outputs the value of $\pi_k(T)$ for a tree T. According to the algorithm, we give some edge deletion rules that result in an optimal k-path cover for a tree T in polynomial time, which is in fact equivalent to the results of Yan et al [17], and the detailed proof is omitted.

However, in order to find an optimal k-path cover for a cactus, we introduce the so-called *Steiner version* of the problem, i.e., *the Steiner k-path cover problem*, defined as follows: Let D and S be two subsets of $V(G)$ such that $D \cup S = V(G)$ and $D \cap S = \emptyset$. The problem is to find the minimum number of vertex disjoint k-paths which cover all the vertices of D. This concept is somehow motivated by the concept of Steiner trees. Clearly, the Steiner k-path cover problem is exactly the k-path cover problem when $D = V(G)$. The set S is called the *Steiner vertex set.* It is easy to see that an optimal Steiner k-path cover may contain vertices of S other than those to be covered. Denote by $\pi_k(G, D)$ the minimum number of vertex disjoint k-paths which cover all the vertices of D, and simply call it the *covering number.*

Like the algorithm in [10], the algorithm for the Steiner k-path cover problem basically operates by applying two types of rules, namely, the edge deletion rules and the cycle opening rules. The cycle opening rules find the edges on cycles that can be deleted from a given cactus without affecting its covering number. The edge deletion rules given in Section 2 construct an optimal k-path cover for a given cactus by decomposing it into two components and covering each component separately.

The rest of the paper is organized as follows. In Section 2, we present the edge deletion rules for the optimal Steiner k-path cover for trees. The cycle opening rules are presented in Section 3. The algorithm for the Steiner k-path cover problem for cacti is developed in Section 4. The last section presents the concluding remarks.

2 The edge deletion rules

In this section we focus on the edge deletion rules. Note that the rules in this section are equivalent to the recursive formulas and the algorithm of Yan et al [17], and so we do not give their detailed proofs. Unlike the algorithm of Yan et al [17], these rules output an optimal Steiner k-path cover for a rooted tree in polynomial time, not only the minimum number. Our algorithm for the Steiner k-path cover problem for trees is based on these edge deletion rules. These rules are repeatedly applied to a forest F, which is initially the tree T. When each component of F is a k-path, the algorithm stops and the union of the components of the current F consists of an optimal Steiner k-path cover of T. The covering number is preserved when edges are deleted according to the rules. It is worth noting that the edge deletion rules in [2, 10] are not applicable here. At first, we give some definitions and notations.

Definition 2.1 *Let S_G be a Steiner k-path cover of a graph G, we say that S_G* employs an edge *$e \in E(G)$ if some path of S_G contains e. A path $v_1v_2\cdots v_m$ is called a* pendant path starting at *v_1, where $m \geq 2$, $d(v_i) = 2$, $1 < i < m$, and $d(v_m) = 1$. Let T be a rooted tree. A vertex with at least two sons in T is called a* fork *if each of its descendants is of degree at most 2. Clearly, each fork ($\neq$ root) in T is of degree at least 3.*

We have the following results.

Lemma 2.1 *Let T be a rooted tree. If there is a vertex x of degree 1 such that $x \in S$, then $\pi_k(T, D) = \pi_k(T - x, D)$.*

Proof. Let S_T be an optimal Steiner k-path cover of the tree T. If the vertex x is not covered by S_T, then we are done. If x is covered by S_T, denote by p the path in S_T which covers the vertex x. It is easy to see that x is an end vertex of the path p. Let $p^* = p - x$. Then $S_T - p + p^*$ is an optimal Steiner k-path cover of both T and $T - x$, in which x is not covered. This completes the proof. ∎

Lemma 2.2 *Let v_1 be a fork in the rooted tree T, where each leaf is a vertex of D. If there is a pendant path $v_1v_2\cdots v_m$ starting at v_1 such that $m > k$, then $\pi_k(T, D) = \pi_k(T - v_{m-k}v_{m-k+1}, D)$.*

Proof. Let S_T be an optimal Steiner k-path cover of T. Suppose that S_T employs the edge $v_{m-k}v_{m-k+1}$. Let $p = p_1v_{m-k}v_{m-k+1}p_2$ be the k-path in S_T that employs the edge $v_{m-k}v_{m-k+1}$. Then the path $p_3 = v_{j+1}v_{j+2}\cdots v_m$ in the remaining part must be a k-path in S_T. It is easy to see that $v_{m-k+1}p_2p_3$ is a k-path in T. Then $\overline{S_T} = S_T - p - p_3 + p_1v_{m-k} + v_{m-k+1}p_2p_3$ is a desired optimal Steiner k-path cover of both T and $T - v_{m-k}v_{m-k+1}$. This completes the proof. ∎

So, in the following we can assume that each leaf is a vertex of D and each pendant path contains at most k vertices in T. We have the following results.

Lemma 2.3 *Let v_1 be a fork in the rooted tree T, where each leaf is a vertex of D. Suppose that each pendant path starting at v_1 contains at most k vertices in T. If there are two pendant paths, $v_1v_2\cdots v_m$ and $v_1v_2^*\cdots v_n^*$, whose union contains at most k vertices, then $\pi_k(T,D)=\pi_k(T-\{v_1x\in T: x\neq v_2, x\neq v_2^*\},D)$.*

Proof. Let S_T be an optimal Steiner k-path cover of T. Suppose that S_T employs at most one of the edges v_1v_2 and $v_1v_2^*$. Let $p=p_1v_1p_2$ be the k-path in S_T containing v_1, where p_i, $i=1,2$, may contain no vertices.

Case 1. Without loss of generality, we assume that only v_1v_2 is employed by S_T, or say that v_2 is contained in p_1. Then $p_3=v_2^*\cdots v_n^*$ is a k-path in S_T and $p_1=v_m\cdots v_2$. Then $\overline{S_T}=S_T-p-p_3+p_2+p_1v_1p_3$ is a desired optimal Steiner k-path cover of both T and $T-\{vx\in T: x\neq v_2, x\neq v_2^*\}$.

Case 2. Suppose that none of the edges v_1v_2 and $v_1v_2^*$ is employed by S_T. Then both $p_1=v_2\cdots v_m$ and $p_2=v_n^*\cdots v_2^*$ are k-paths in S_T. Let $p=p_3v_1p_4$ be the k-path in S_T containing v_1, where p_i, $i=3,4$, contains at least one vertex. Then $\overline{S_T}=S_T-p-p_1-p_2+p_3+p_4+p_2v_1p_1$ is a desired optimal k-path cover of both T and $T-\{vx\in T: x\neq v_2, x\neq v_2^*\}$. This completes the proof. ∎

Lemma 2.4 *Let v_1 be a fork in the rooted tree T, where each leaf is a vertex of D. Suppose that there are several pendant paths starting at v_1, each of which contains at most k vertices. Let $u_1,u_2,\cdots,u_m$ be all the sons of v_1 in T and let $v_1u_1\cdots x$ be a pendant path with the minimum number of vertices among all pendant paths starting at v_1. If the union of any two pendant paths starting at v_1 contains more than k vertices, then $\pi_k(T,D)=\pi_k(T-\{v_1u_i: i=2,3,\cdots,m\},D)$.*

Proof. Let S_T be an optimal Steiner k-path cover of T. Clearly, any single k-path of S_T cannot cover two pendant paths starting at v_1 completely. Suppose that S_T employs the edge v_1u_i for some $i\geq 2$. Let $p=p_1v_1u_ip_2$ be a k-path in S_T, which employs the edge v_1u_i. Without loss of generality, we can assume that $v_1u_ip_2$ is one of the pendant paths starting at v_1. We distinguish the following two cases.

Case 1. If p_1 contains no descendants of v_1. Then $p_3=u_1\cdots x$ must be a k-path in S_T. It is easy to see that $\overline{S_G}=S_G-p-p_3+p_1v_1p_3+u_ip_2$ is the desired optimal Steiner k-path cover of both T and $T-\{v_1u_i: i=2,3,\cdots,m\}$.

Case 2. Otherwise, we can assume that $p_1=y'\cdots u_j$. Let $v_1u_j\cdots y'y''\cdots y$ be a pendant path starting at v_1. Then $p_4=y\cdots y''$ is a k-path in S_T.

Subcase 2.1. If $j\neq 1$, then $p_3=u_1\cdots x$ is a k-path in S_T. It is easy to see that $\overline{S_G}=S_G-p-p_3-p_4+p_4p_1+v_1p_3+u_ip_2$ is the desired optimal Steiner k-path cover of both T and $T-\{v_1u_i: i=2,3,\cdots,m\}$.

Subcase 2.2. If $j=1$, then $x=y$ and $p_4=x\cdots y''$. Then $\overline{S_G}=S_G-p-p_4+p_4p_1v_1+u_ip_2$ is the desired optimal Steiner k-path cover of both T and $T-\{v_1u_i: i=2,3,\cdots,m\}$. ∎

Based on the previous lemmas, we can easily develop an efficient algorithm for the Steiner k-path cover problem of trees. Note that our goal is to find an optimal Steiner k-path cover for cacti, and a cactus is a tree if it contains no cycles. For technical reasons, we need some more definitions and notations.

Definition 2.2 *Let G be a cactus and v be a vertex in a cycle of G. Let G_i, $i \in I$, be the components of $G - v$ such that $G[V(G_i) \cup v]$ contains no cycles. The tree $T_v = \bigcup_{i \in I} G[V(G_i) \cup v]$, rooted at v, is called the tree* suspended at *v. Note that, if the index set I is empty, T_v consists of only the single vertex v.*

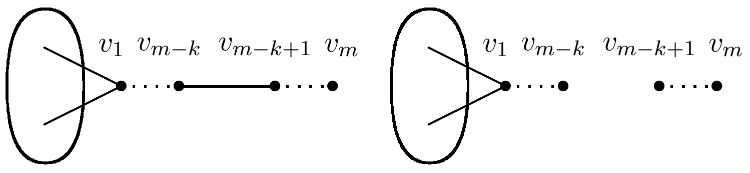
Before opening the cycles of the cactus G, we need to trim the suspended tree T_v for each vertex v in a cycle of G. Similarly to trees, we have the following results.

Lemma 2.5 *Let G be a cactus. If there is a vertex x of degree one such that $x \in S$, then $\pi_k(G, D) = \pi_k(G - x, D)$.*

Proof. The proof is analogous to that of Lemma 2.1, and the details are omitted. ∎

Lemma 2.6 *Let G be a cactus such that each vertex of degree 1 belongs to the set D. Let v_1 be a fork in the suspended tree T_v in G. If there is a pendant path $v_1 v_2 \cdots v_m$ starting at v_1 such that $m > k$ in G, then $\pi_k(G, D) = \pi_k(G - v_{m-k}v_{m-k+1}, D)$.*

Proof. The proof is analogous to that of Lemma 2.2, and the details are omitted. ∎



Before edge deletion After edge deletion

Figure 1

The edge deletion rule of Lemma 2.6 is illustrated in Figure 1. It is easy to see that, based on Lemma 2.6, we can separate k-paths from G, such that each pendant path starting at the fork v_1 in the current cactus has at most k vertices. Furthermore, we have the following result.

Lemma 2.7 *Let G be a cactus such that each vertex of degree 1 belongs to the set D. Let v_1 be a fork in the suspended tree T_v in G. Suppose that each pendant path starting at v contains at most k vertices. If there are two pendant paths, $v_1 v_2 \cdots v_m$ and $v_1 v_2^* \cdots v_n^*$, whose union contains at most k vertices, then $\pi_k(G, D) = \pi_k(G - \{v_1 x \in G : x \neq v_2, x \neq v_2^*\}, D)$.*

Proof. The proof is analogous to that of Lemma 2.3, and the details are omitted. ∎

The edge deletion rule of Lemma 2.7 is illustrated in Figure 2. Note that, even when v_1 lies in some cycles in G, the edge deletion rule of Lemma 2.6 is still applicable. Then the edge deletion rule of Lemma 2.7 may also be used to open cycles in G. According to Lemmas 2.6 and 2.7, we can separate k-paths from the cactus G. And, when no such pendant paths as those in Lemmas 2.6 and 2.7 exist for the fork v_1 in the current cactus, each of the pendant paths starting at v_1 contains at most k vertices, and the union of any two of them contains more than k vertices. For this case, we have the following result.

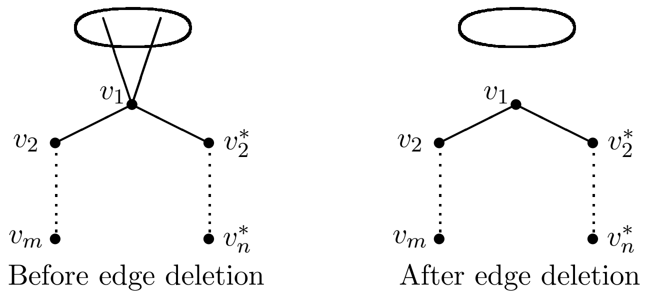
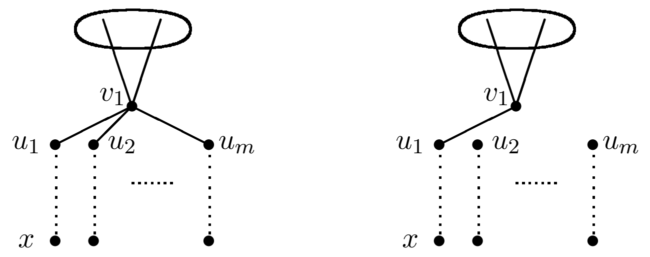


Figure 2

Lemma 2.8 *Let G be a cactus such that each vertex of degree 1 belongs to the set D. Let v_1 be a fork in the suspended tree T_v in G, starting at which there are several pendant paths each of which contains at most k vertices. Let $u_1, u_2, \cdots, u_m$ be all the sons of v_1 in T_v and $v_1u_1 \cdots x$ be a pendant path with the minimum number of vertices among all pendant paths starting at v_1 in T_v. If the union of any two pendant paths starting at v_1 contains more than k vertices, then $\pi_k(G, D) = \pi_k(G - \{v_1u_i : i = 2, 3, \cdots, m\}.D)$.*

Proof. The proof is analogous to that of Lemma 2.4, and the details are omitted. ∎

The edge deletion rule of Lemma 2.8 is illustrated in Figure 3. Lemma 2.8 implies that we can delete all the edges v_1u_i, $i \geq 2$, while preserving the covering number of the cactus. According to the previous lemmas, each suspended tree in the current cactus consists of either a single vertex or a single pendant path with at most k vertices in the current cactus. We present a simple outline of these rules as follows.



Before edge deletion After edge deletion

Figure 3

Procedure: Trim G.

Step 1. If there is a vertex x of degree 1 in G such that $x \in S$, then delete the vertex x.

Step 2. Let u be a fork in some suspended tree in the current cactus.

2.1. If there is a pendant path starting at u which contains more than k vertices, then trim it by the edge deletion rule of Lemma 2.6.

2.2. If there are two pendant paths starting at u whose union contains at most k vertices, then trim it by the edge deletion rule of Lemma 2.7.

2.3. If there are at least two pendant paths starting at u, among which the union of any two pendant paths contains more than k vertices, then trim it by the edge deletion rule of Lemma 2.8.

3 The cycle opening rules

3.1 The end cycles

In Section 2, Lemma 2.7 gives a rule to open cycles in cacti. But, it is not enough for us. In this section we consider cacti to which none of the previous rules is applicable. Then each suspended tree in the cactus consists of either a single vertex or a single pendant path with at most k vertices. We focus our discussion on these cacti and give rules to open the cycles of them. It is worth noting that we only choose some particular cycles to open. First we introduce some more necessary definitions and notations.

Definition 3.1 *A cactus is called a trimmed cactus if each suspended tree of it consists of either a single vertex or a single pendant path with at most k vertices. Let G be a cactus and C be a cycle of G. Denote by C_G the union of the cycle C and all the trees suspended at vertices on C. Obviously, C_G is an induced subgraph of G and contains a unique cycle. A cycle C of G is called an end cycle if $G = C_G$, or there is a unique vertex v on C which is adjacent to a vertex in $V(G) \setminus V(C_G)$. If $G = C_G$, any vertex v on C is* *called an anchor of C; otherwise, the unique vertex v on C which is adjacent to a vertex in $V(G) \setminus V(C_G)$ is called the anchor of C.*

Note that, though we adopt the same terms as those in [10], it is not the same thing in essence. The concept of end cycles plays a key role in this paper. If G is unicyclic, then the unique cycle is an end cycle in G. Suppose that G is a cactus containing at least two cycles, then the existence of end cycles can be proved by constructing a rooted tree T, whose vertex set is the set of cycles in G, as follows. Let C_0 be a cycle in G, which is regarded as the root of T. Then C is one of the sons of C_0 in T if and only if there is a path P from C_0 to C such that $V(P) - V(C) \cup V(C_0)$ does not contain any vertex in other cycles in G. Label the cycle C where $C \in N_T[C_0]$. Let C_i be labelled, then an unlabelled vertex C_j is one of the sons of C_i in T if and only if there is a path P from C_i to C_j such that $V(P) - V(C_i) \cup V(C_j)$ does not contain any vertex in other cycles in G. Label the cycle C where $C \in N_T[C_i]$. The other cycles of G can be labelled similarly. It is easy to see that $C \neq C_0$ is an end cycle in G if and only if C is a leaf in T.

Next we show how the end cycles in the trimmed cactus can be opened while the covering number of the cactus is preserved. The basic motivation is that, for any cactus containing cycles, any optimal Steiner k-path cover does not employ at least one edge on each cycle. It is easy to see that, if the given cactus contains a unique cycle, it is not difficult to find an optimal Steiner k-path cover of it. A simple and efficient method is to find an optimal Steiner k-path cover of the tree obtained from deleting an edge of the cycle. After doing this for each edge of the cycle, the cover with the minimum number of paths is an optimal Steiner k-path cover of the given cactus. Unfortunately this simple method does not run efficiently when the given cactus contains several cycles.

From now on we focus on the trimmed cacti with several cycles. Let G be a such cactus and C be an end cycle in G, whose anchor is the vertex v. Let e be an edge on C. Then the cactus $G-e$ may be not trimmed, and the only possibility is that the suspended tree T_v of it needs to be trimmed. But the covering number of G may not be preserved, i.e., $\pi_k(G, D) \neq \pi_k(G - e, D)$ for some $e \in C$. For example, let G be the union of a cycle and a path with a unique common vertex v, $D = V(G)$ and $k = |V(G)|$, then $\pi_k(G, D) \neq \pi_k(G - e, D)$ for any edge $e \in C - v$. Note that the root v has at most three sons in the suspended tree T_v in $G - e$. As illustrated in the following procedure, we trim the suspended tree T_v in the cactus $G - e$ by the rules of Lemmas 2.5 through 2.8 under some additional restrictions.

Procedure$_{(C,e)}$.

Step 1. If there is a vertex x of degree 1 in G such that $x \in S$, then delete the vertex x.

Step 2. Let u be a fork in the suspended tree T_v in the current cactus.

2.1. If there is a pendant path starting at u which contains more than k vertices, then trim it by the edge deletion rule of Lemma 2.6.

2.2. If $u \neq v$ and there are two pendant paths starting at u whose union contains at most k vertices, then trim it by the edge deletion rule of Lemma 2.7.

2.3. If there are at least two pendant paths starting at u, among which the union of any two pendant paths contains more than k vertices, then trim it by the edge deletion rule of Lemma 2.8.

2.4. If there are exactly three pendant paths starting at v in the current cactus, the union of some two of which contains at most k vertices, then let $vx \cdots y$ be the largest pendant path among the three, and delete the edge vx.

It is easy to see that in each step we have already separated some k-paths from the current cactus. Denote by $S_e(C)$ the set of all the k-paths separated in Procedure$_{(C,e)}$. Finally, we obtain a new cactus, denoted by G_e, in which the suspended tree T_v consists of either a single vertex, or a pendant path with at most k vertices, or two pendant paths whose union contains at most k vertices. Since $C_G - e$ is a tree rooted at the vertex v, according to the rules in Section 2, it is easy to see that

$$\pi_k(C_G - e, D \cap V(C_G)) = \begin{cases} |S_e(C)|, & \text{if } v \in S \text{ and the suspended tree } T_v \text{ in } G_e \text{ consists of only the vertex } v; \\ |S_e(C)| + 1, & \text{otherwise.} \end{cases}$$

Based on the three possible cases of the suspended tree T_v in the cactus G_e for each e, we partition the edges of C into three different classes as follows:

$C_1 = \{e \in C$: the suspended tree T_v in G_e consists of only the vertex $v\}$,

$C_2 = \{e \in C$: the suspended tree T_v in G_e consists of only a pendant path with at most k vertices$\}$,

$C_3 = \{e \in C$: the suspended tree T_v in G_e consists of only two pendant paths whose union contains at most k vertices$\}$.

Then, $C_1 \cup C_2 \cup C_3 = E(C)$ and $C_i \cap C_j = \emptyset$, $i \neq j$. For $i = 1, 2, 3$, let

$$s_i = \begin{cases} \min_{e \in C_i}\{|S_e(C)|\}, & \text{if } C_i \neq \emptyset; \\ \infty, & \text{if } C_i = \emptyset. \end{cases}$$

An obvious consequence is that

$$\pi_k(C_G, D \cap V(C_G)) = \begin{cases} \min\{s_1 + 1, s_2 + 1, s_3 + 1\}, & \text{if } v \in D; \\ \min\{s_1, s_2 + 1, s_3 + 1\}, & \text{if } v \in S. \end{cases}$$

And, by simple arguments we have

$$\pi_k(C_G - v, D \cap V(C_G - v)) = \min\{s_1, s_2 + 1, s_3 + 2\}.$$

Without loss of generality, we assume that $|S_{e_i}(C)| = s_i$, $e_i \in C_i$ if $s_i \neq \infty$, and the single pendant path starting at v in G_{e_2} contains as few vertices as possible if $s_2 \neq \infty$.

3.2 Opening end cycles

In this subsection we present a rule to choose an edge among the edges e_i of an end cycle C in order to open the cycle, while the covering number is preserved. Before presenting the cycle opening rules, we need some preparations. Consider the subgraph C_G rooted at v. A *v-rooted Steiner k-path cover* of C_G is a Steiner k-path cover in which v is an end vertex of a path. The *v-rooted Steiner k-path cover number*, denoted by $\pi_k(C_G|v, D \cap V(C_G))$, is the number of paths in a v-rooted Steiner k-path cover of C_G with minimum cardinality. Denote by $l_k(C_G, v)$ the minimum number of vertices in a path containing v in a v-rooted Steiner k-path cover of size $\pi_k(C_G|v, D \cap V(C_G))$. Note that the root vertex v is covered in any optimal v-rooted Steiner k-path cover of C_G whether $v \in D$ or $v \in S$.

According to the edge deletion rules in Section 2, and the algorithm and recursive formulas of Yan et al [17], we have the following result, and the detailed proof is omitted.

Lemma 3.1 *If $min\{s_1-1, s_3\} \geq s_2$, then $S_{e_2}(C)+q$ is an optimal Steiner k-path cover of C_G such that $|V(q)| = l_k(C_G, v)$, where q is the unique pendant path starting at v in G_{e_2}. And, we have*

$$\pi_k(C_G|v, D \cap V(C_G)) = \begin{cases} s_3+2, & \text{if } min\{s_1-1, s_2\} > s_3, \\ s_2+1, & \text{if } min\{s_1-1, s_3\} \geq s_2, \\ s_1+1, & \text{otherwise.} \end{cases}$$

∎

Then we have the following results.

Lemma 3.2 *If $min\{s_1-1, s_2\} > s_3$, then $\pi_k(G, D) = \pi_k(G - e_3, D)$.*

Proof. Let S_G be an optimal Steiner k-path cover of G and $S_c \subseteq S_G$ be the set of the k-paths covering the vertices of $D \cap V(C_G)$. If $S_c \subseteq C_G$, then S_c is an optimal Steiner k-path cover of C_G, and the result holds obviously since we can replace S_c by another optimal Steiner k-path cover of C_G, i.e., $S_{e_3}(C)$ plus the union of the two pendant paths starting at v in G_{e_3}. If there is a unique k-path $p = p_1yvp_2 \in S_c$ such that $p_1y \in G \setminus C_G$, we distinguish the following two cases.

Case 1. $vp_2 \subseteq C_G$.

Then $S_c - p + vp_2$ is a v-rooted Steiner k-path cover of C_G. Since $\min\{s_1 - 1, s_2\} > s_3$, and S_c must cover all the vertices of $D \cap V(C_G)$, we have $|S_c| \geq s_3 + 2$. Then $S_G - S_c + p_1y + S_{e_3}(C) + q$ is an optimal Steiner k-path cover of both G and $G - e_3$, where q is the union of the two pendant paths starting at v in the cactus G_{e_3}.

Case 2. $p_2 \subseteq G \setminus C_G$.

Then $S_c - p$ covers all the vertices of $D \cap V(C_G - v)$, and so we have that

$$|S_c - p| \geq \pi_k(C_G - v, D \cap V(C_G - v)) \geq s_3 + 2.$$

Then, $S_G - S_c + S_{e_3}(C) + p_1 + p_2 + q$ is an optimal Steiner k-path cover of both G and $G - e_3$, where q is the union of the two pendant paths starting at v in the cactus G_{e_3}. This completes the proof. ∎

Lemma 3.3 *If $min\{s_2, s_3\} > s_1 - 1$, then $\pi_k(G, D) = \pi_k(G - e_1, D)$.*

Proof. Let S_G be an optimal Steiner k-path cover of G and let $S_c \subseteq S_G$ be the set of the k-paths covering the vertices of $D \cap V(C_G)$. If $S_c \subseteq C_G$, then S_c is an optimal Steiner k-path cover of C_G, and the result holds obviously since we can replace S_c by another optimal Steiner k-path cover of C_G, i.e., $S_{e_1}(C)$, plus the path v if $v \in D$. If there is a unique k-path $p = p_1vp_2 \in S_c$ such that $p_1 \in G \setminus C_G$. We distinguish the following two cases:

Case 1. $vp_2 \subseteq C_G$.

Then $S_c - p + vp_2$ is a v-rooted Steiner k-path cover of C_G, which implies that $|S_c| \geq s_1 + 1$. Then $S_G - S_c + p_1v + S_{e_1}(C)$ is an optimal Steiner k-path cover of both G and $G - e_1$.

Case 2. $p_2 \subseteq G \setminus C_G$.

Then $S_c - p$ is an optimal Steiner k-path cover of $C_G - v$, which implies that $|S_c - p| = |S_{e_1}(C)|$. Then $S_G - S_c + p + S_{e_1}(C)$ is an optimal Steiner k-path cover of both G and $G - e_1$. This completes the proof. ∎

Lemma 3.4 *If $min\{s_1 - 1, s_3\} \geq s_2$, then $\pi_k(G, D) = \pi_k(G - e_2, D)$.*

Proof. Let S_G be an optimal Steiner k-path cover of G and let $S_c \subseteq S_G$ be the set of the k-paths covering the vertices of C_G. If $S_c \subseteq C_G$, then S_c is an optimal Steiner k-path cover of C_G, and the result holds obviously since we can replace S_c by another optimal Steiner k-path cover of C_G, i.e., $S_{e_2}(C)$ plus the pendant path starting at v in G_{e_2}. If there is a unique k-path $p = p_1vp_2 \in S_c$ such that $p_1 \in G \setminus C_G$, then it is easy to see that $|S_c| \geq s_2 + 1$. Let vp_3 be the unique pendant path starting at v in G_{e_2}. We distinguish two cases:

Case 1. $vp_2 \subseteq C_G$.

Then $S_c - p + vp_2$ is a v-rooted Steiner k-path cover of C_G. From Lemma 3.2 we have that $|V(vp_3)| \leq |V(vp_2)|$. This implies that $S_G - S_c + S_{e_2}(C) + p_1vp_3$ is an optimal Steiner k-path cover of both G and $G - e_2$.

Case 2. $p_2 \subseteq G \setminus C_G$.

Then $S_c - p$ is an optimal Steiner k-path cover of $C_G - v$, which implies that $|S_c - p| = s_2 + 1$. Then $S_G - S_c + p + S_{e_2}(C) + p_3$ is an optimal Steiner k-path cover of both G and $G - e_2$. This completes the proof. ∎

The only case left is that $s_1 - 1 = s_3 < s_2$. Though we can not determine exactly which of the e_i's should be deleted, we have the following result.

Lemma 3.5 *If $s_1 - 1 = s_3 < s_2$, then $\pi_k(G, D) = \pi_k(G - e_1, D)$ or $\pi_k(G, D) = \pi_k(G - e_3, D)$.*

Proof. Let S_G be an optimal Steiner k-path cover of G and let $S_c \subseteq S_G$ be the set of the k-paths covering the vertices of C_G. If $S_c \subseteq C_G$, then S_c is an optimal Steiner k-path cover of C_G, and the result holds obviously since we can replace S_c by another optimal Steiner k-path cover of C_G, i.e., $S_{e_3}(C)$ plus the union of the two pendant paths starting at v in G_{e_3}. If there is a unique k-path $p = p_1vp_2 \in S_c$ such that $p_1 \in G \setminus C_G$. We distinguish two cases:

Case 1. $p_2 \subseteq G \setminus C_G$.

Then $S_c - p$ is an optimal Steiner k-path cover of $C_G - v$, which implies that $|S_c - p| = s_1$. Then $S_G - S_c + p + S_{e_1}(C)$ is an optimal Steiner k-path cover of both G and $G - e_1$.

Case 2. $p_2 \subseteq C_G$.

Then $S_c - p + vp_2$ is a v-rooted Steiner k-path cover of C_G, which implies that $|S_c| \geq s_1 + 1$. Then $S_G - S_c + p_1 v + S_{e_1}(C)$ is an optimal Steiner k-path cover of both G and $G - e_1$. This completes the proof. ∎

From the previous lemmas, we can open the end cycle C by comparing the values of $s_1 - 1$, s_2 and s_3, except for the case $s_1 - 1 = s_3 < s_2$. According to Lemma 3.6, we can delete either e_1 or e_3 to open the cycle C if $s_1 - 1 = s_3 < s_2$, but to determine exactly which one to be deleted is still a problem. For convenience, we label the anchor v of the end cycle C by $l(C)$ if $s_1 - 1 = s_3 < s_2$. Let $G^{'} = G_{e_1}$ and $D^{'} = (D - v) \cap V(G^{'})$. We have the following result.

Lemma 3.6 $\pi_k(G, D) = \pi_k(G^{'}, D^{'}) + s_1$.

Proof. Let $S_{G'}$ be an optimal Steiner k-path cover of $G^{'}$, where $V(G^{'}) \setminus D^{'}$ is the Steiner vertex set. It is easy to see that the k-path cover $S_{G'} \cup S_{e_1}(C)$ (or, $S_{G'} \cup S_{e_3}(C)$) is an optimal Steiner k-path cover of G if v is covered (or, not covered) in $S_{G'}$. ∎

Then, recursively, we can find an optimal Steiner k-path cover of the cactus G easily.

4 Algorithm

Based on our rules, we are ready now to present an efficient algorithm for finding an optimal Steiner k-path cover of a given cactus G. It is easy to see that it can find an optimal k-path cover for the given cactus G if the Steiner vertex set $S = \emptyset$.

Algorithm: Find an optimal Steiner k-path cover of a cactus.

Input: A cactus G with the Steiner vertex set S, and a positive integer $k > 0$.

Output: An optimal Steiner k-path cover of G.

Step 1. Initial state: $S_G = \emptyset$, $L = \emptyset$ and $S_e(C) = \emptyset$ for each edge e on each cycle C.

Step 2. Do **Procedure: Trim** G to trim G, and

$$L \leftarrow L \cup \{x : x \text{ is labelled and deleted in Step 1 of } \textbf{Procedure: Trim } G\}.$$

Step 3. If every component of G is a k-path, go to Step 6.

Step 4. Let C be an end cycle in G, whose anchor is the vertex v.

```
        4.1. For each edge $e \in C$, do $\textbf{Procedure}_{(C,e)}$ and update the set
             $S_e(C)$ for each edge $e \in C$.
        4.2. Determine the value of $s_i$ and the corresponding edges $e_i$, $i = 1, 2, 3$, on $C$.
Step 5. If $\min\{s_1 - 1, s_2\} > s_3$, do $G \leftarrow G - e_3$, then go to Step 2.
Step 6. If $\min\{s_2, s_3\} > s_1 - 1$, do $G \leftarrow G - e_1$, then go to Step 2.
Step 7. If $\min\{s_1 - 1, s_3\} \geq s_2$, do $G \leftarrow G - e_2$, then go to Step 2.
Step 8. If $s_1 - 1 = s_3 < s_2$, do
        label the vertex $v$ by $l(C)$, $G \leftarrow G_{e_1}$, $D \leftarrow (D - v) \cap V(G_{e_1})$, and
        $S \leftarrow (V(G_{e_1}) \cap S) \cup \{v\}$, then go to Step 2.
Step 9. For each vertex $x \in L$, which is labelled by $l(C)$, do
        $G \leftarrow G \cup S_{e_3}(C)$.
        For each vertex $y \notin L$, which is labelled by $l(C)$, do
        $G \leftarrow G \cup S_{e_1}(C)$.
  Stop!
```

Theorem 4.1 *Given a cactus G, our* **Algorithm** *produces an optimal Steiner k-path cover of G in polynomial time, and the complexity is upper bounded by $O(|V(G)|^2)$.*

Proof. The correctness of the theorem is obvious from the previous results. By results in [17], both **Procedure: Trim** G and $\textbf{Procedure}_{(C,e)}$ run in linear time. Then for each end cycle C, Step 4 can be executed in $|V(C)| \bullet O(|V(C_G)|)$ time. For each end cycle C, Step 2 can be executed in $O(|V(G)|)$ time, and Steps 5 through 9 can be executed in a constant time. Since there are at most $O(|V(G)|)$ cycles in G, it is easy to see that our **Algorithm** can be executed in $O(|V(G)|^2)$ time. This completes the proof. ∎

5 Conclusions

In this paper we introduce the so-called Steiner version of the k-path cover problem for graphs. The problem is NP-complete since it is a generalization of the k-path cover problem. Motivated by the intractability and notable applications of the k-path cover problem, the k-path cover problem has been well studied. But there are only few results restricted on special classes of graphs. In this paper we presented a polynomial time algorithm for the Steiner version of the k-path cover problem for cacti, which not only finds the minimum number but also gives an optimal k-path cover.

Acknowledgement. The authors are very grateful to the referees for helpful suggestions and comments.

References

[1] S.R. Arikati and C.P. Rangan, Linear algorithm for optimal path cover problem on interval graphs, Inform. Process. Lett. 35(1990), 149-153.

[2] F.T. Boesch, S. Chen and J.A.M. McHugh, On covering the points of a graph with point disjoint paths, Proc. 1973 Capital Conf. on Graph Theory and Combinatorics (1974), 201-212.

[3] F.T. Boesch and J.F. Gimpel, Covering the points of a digraph with point disjoint paths and its application to code optimaization, J. ACM 24(1977), 192-198.

[4] M.A. Bonuccelli and D.P. Bovet, Minimum node disjoint path covering for circular-arc graphs, Inform. Process. Lett. 8(1979), 159-161.

[5] G.J. Chang and D. Kuo, The L(2,1)-labeling problem on graphs, SIAM J. Discrete Math. 9(1996), 309-316.

[6] M.R. Garey and D.S. Johnson, Computers and Intractability, W.H. Freeman, San Francisco, 1979.

[7] M.C. Golumbic, Algorithmic Graph Theory and Perfect Graphs, Academic Press, New York, 1980.

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

[9] J. Misra and R.E. Tarjan, Optimal chain partitions of trees, Inform. Proces. Lett. 4(1975), 24-26.

[10] S. Moran and Y. Wolfstahl, Optimal covering of cacti by vertex-disjoint paths, Theoret. Comput. Sci. 84(1991), 179-197.

[11] S. Pinter and Y. Wolfstahl, On mapping processes to processors, Internat. J. Parallel Programming 16(1987), 11-16.

[12] R. Srikant, R. Sundaram, K.S. Singh and C.P. Rangan, Optimal path cover problem on block graphs and bipartite permutation graphs, Theoret. Comput. Sci. 115(1993), 351-357.

[13] G. Steiner, On the k-path partition problem in cographs, Cong. Numer. 147(2000), 89-96.

[14] G. Steiner, On the k-path partition of graphs, Theoret. Comput. Sci. 290(2003), 2147-2155.

[15] A.S. Tanenbaum, Computer Networks, Prentice-Hall, Englewood Cliffs, 1981.

[16] J.H. Yan and G.J. Chang, The path partition problem, Inform. Process. Lett. 52(1994), 317-322.

[17] J.H. Yan, G.J. Chang, S.M. Hedetniemi and S.T. Hedetniemi, k-path partitions in trees, Discrete Appl. Math. 78(1997), 227-233.