OPTIMAL COVERING OF CACTI
BY VERTEX-DISJOINT PATHS

by

Shlomo Moran and Yaron Wolfstahl

Technical Report #501
March 1988
Abstract

A path cover (abbr. cover) of a graph $G$ is a set of vertex-disjoint paths which cover all the vertices of $G$. An optimal cover of $G$ is a cover of the minimum possible cardinality. The optimal covering problem is known to be NP-complete even for cubic 3-connected planar graphs where no face has fewer than 5 edges. Motivated by the intractability of this problem, we develop an efficient optimal covering algorithm for cacti (i.e. graphs where no edge lies on more than one cycle). In doing so we generalize the results of [2] and [9], where optimal covering algorithms for trees and graphs where no two cycles share a vertex were presented.
1. Introduction

Let $G=(V_G, E_G)$ be an undirected graph with no self loops or parallel edges. A path in $G$ is either a single vertex $v \in V_G$ or a sequence of distinct vertices $(v_1, v_2, \ldots, v_k)$ where for $1 \leq i < k-1$, $(v_i, v_{i+1}) \in E_G$. A path cover (abbreviated as cover) of $G$ is a set of vertex-disjoint paths which cover all the vertices of $G$. An optimal cover of $G$ is a cover of the minimum possible cardinality. The cardinality of such a cover is called the covering number of $G$, and is denoted by $\pi(G)$.

The concept of graph covering has many practical applications. For example, in order to establish ring protocols [10], a computer network may be augmented by some auxiliary edges so as to make it Hamiltonian [5]. It is easily verified that the minimum number of additional edges needed to make a network Hamiltonian is identical to the covering number of the network. Other notable applications of graph covering are code optimization [3] and mapping parallel programs to parallel architectures [9].

The problem of finding an optimal cover is NP-complete even for cubic 3-connected planar graphs where no face has fewer than 5 edges [6]. There are, however, several results on optimal covering of restricted classes of graphs. Boesch, Chen and McHugh have derived in [2], among other things, an optimal covering algorithm for trees. Their result was generalized by Pinter and Wolfstahl [9], who developed an efficient optimal covering algorithm for graphs where no two cycles share a vertex. Boesch and Gimpel [3] have considered the related problem of covering a directed acyclic graph by directed paths.

The main result presented in this paper generalizes the above results of [2] and [9]. Specifically, we develop a linear optimal covering algorithm for cacti, that is, graphs where no edge lies on more than one cycle [1,8,11] (see Figure 1). We note that the class of cacti properly contains the graph classes considered in [2] and [9]. The algorithm basically operates by applying two types of rules, namely, edge-deletion rules and a recursive decomposition rule. The edge-deletion rules characterize the edges that can be deleted from a given cactus without affecting its covering number. The recursive decomposition rule provides a tool for constructing an optimal cover of a cactus by decomposing it into two components and covering each component separately. We believe that the combined use of the two types of rules is a feature of independent interest.

The rest of this paper is organized as follows. The edge-deletion rules are presented in Section 2. The recursive decomposition rule is presented in Section 3. The algorithm, developed in Section 4, specifies the order by which those rules are to be applied.
2. Edge-Deletion Rules

The edge-deletion rules are presented in the lemmas below. First, we need some definitions.

Definition 1: Let $S_G$ be a cover of a graph $G = (V_G, E_G)$. We say that $S_G$ employs an edge $e \in E_G$ if some path in $S_G$ includes $e$. A trail starting at $v_1 \in V_G$ is a path $(v_1, v_2, \ldots, v_k)$ containing two or more vertices, where $\deg(v_i) = 2$ for $1 < i < k$ and $\deg(v_k) = 1$. If exactly one trail starts at $v_1$ then this trail is denoted by $tr(v_1)$. A vertex $v \in V_G$ is a fork if $\deg(v) \geq 3$ and at least two trails start at $v$. A vertex $v_1 \in V_G$ is a semi-fork if $\deg(v_1) \geq 3$, exactly one trail $(v_1, v_2, \ldots)$ starts at $v_1$, and $v_1$ is adjacent to a vertex $w$ ($w \neq v_2$) where $\deg(w) = 2$. A trimmed cactus is a cactus containing no forks.

The following proposition is often used in the sequel.

Proposition 1: Let $S$ be a cover of a graph $G = (V_G, E_G)$. Let $u, v$, and $w$ be vertices in $V_G$ where

1. $\{(u,v),(u,w)\} \subseteq E_G$,
2. $(u, w)$ is employed by $S$ but $(u, v)$ is not, and
3. $v$ is an end-vertex of some path in $S$.

Then there exists a cover of $G$, denoted by $S'$, that employs $(u, v)$ but not $(u, w)$ and satisfies $|S'| \leq |S|$.

Proof: Straightforward.

Lemma 1 below is easily proved using the above proposition. This lemma can be used to convert a given cactus into union of trimmed cacti.

Lemma 1 [9] (Deletions due to forks and semi-forks): Let $G = (V_G, E_G)$ be a cactus. Let $v_1 \in V_G$ be a vertex of degree 3 or more which is the start-point of a trail $(v_1, v_2, \ldots, v_k)$ and is adjacent to a vertex $w \neq v_2$ of degree 1 or 2. Then $G' = (V_G, E'_G)$ where $E'_G = E_G - \{(x,v_1) \mid (x,v_1) \in E_G, x \notin \{v_2,w\}\}$ satisfies $\pi(G) = \pi(G')$.

Next, we define the concept of end-cycle which plays a key role in the development of our covering algorithm. In fact, the rest of the edge-deletion rules, as well as the recursive decomposition rule, are applicable to end-cycles. The formal definition of end-cycle is preceded by some necessary additional definitions.

Definition 2: Let $G = (V_G, E_G)$ be a connected graph. Let $A = (V_A, E_A)$ and let $B = (V_B, E_B)$ be edge-disjoint connected subgraphs of $G$. A vertex $u \in V_G$ separates $A$ from $B$ if for all $v \in V_A$ and $w \in V_B$, all the paths connecting $v$ and $w$ contain $u$. If $u \in V_A$ and $u$ separates $A$ from the subgraph induced by $V_G - V_A$, we say that $u$ separates $A$ from $G$. In this case, we also say that $u$ is a separating vertex. The set of connected components separated from $G$ by a vertex $u$ is denoted by $CC(u)$.

A crown is a connected graph containing a single cycle which satisfies
At least one vertex on the cycle is of degree 2.

Each vertex \( u \) on the cycle is either of degree 2 or of degree 3. In the latter case, \( u \) is the start-point of a trail.

Given a crown \( C \), the unique cycle in \( C \) is denoted by \( C^o \), and the degree of each vertex \( v \) in \( C \) is denoted by \( \deg_c(v) \).

Let \( C \) be a crown that is a proper subgraph of a cactus \( G \). We say that \( C \) is an end-cycle of \( G \) (denoted \( C \sim G \)) if the following hold:

1. There exists a vertex \( u \) on \( C^o \) such that \( \deg_c(u)=2 \) and \( u \) separates \( C \) from \( G \). This vertex is called the anchor of \( C \).
2. If \( u \) belongs to no cycle other than \( C^o \), then there is a vertex \( v \) not on \( C \), such that \( v \) separates \( C \) from all cycles in \( G \) (see crown \( C_1 \) in Figure 2).
3. If \( u \) belongs to cycles other than \( C^o \), then all the connected components in \( CC(u) \), except perhaps one, contain at most one cycle (see crown \( C_2 \) in Figure 2).

Let \( C \) be an end-cycle of \( G \). If each vertex \( v \) on \( C^o \) satisfies \( \deg_c(v)=2 \), then \( C \) is an end-cycle of order 2. If each vertex \( v \) on \( C^o \) (except for the anchor) satisfies \( \deg_c(v)=3 \), then \( C \) is an end-cycle of order 3. For example, crown \( C_1 \) in Figure 2 is an end-cycle of order 2, while crown \( C_2 \) is an end-cycle of order 3.

(Insert Figure 2 here)

Next, we prove that an end-cycle must exist in any trimmed cactus that properly contains a cycle.

**Definition 3:** If \( G=(V_G, E_G) \) has a separation vertex then it is called separable, and it is called nonseparable otherwise.

Let \( V' \subseteq V_G \). The subgraph induced by \( V' \) is called a nonseparable component of \( G \) if it is nonseparable and if for any larger \( V'' \), \( V' \subseteq V'' \subseteq V_G \), the subgraph induced by \( V'' \) is separable. Let \( N = \{n_1, n_2, \ldots, n_k\} \) be set of the nonseparable components of \( G \), and \( S = \{s_1, s_2, \ldots, s_p\} \) be the set of its separating vertices. The superstructure of \( G \) is the tree \( T=(V_T, E_T) \), where \( V_T = N \cup S \) and \( E_T = \{(s_i, n_j) \mid s_i \text{ is a vertex of } n_j \} \) [4]. For each \( v \in V_T \), let \( g(v) \) denote the nonseparable component or the separation vertex in \( G \) which corresponds to \( v \).

**Lemma 2 (Existence of end-cycles):** If a trimmed cactus, \( G=(V_G, E_G) \), properly contains a cycle, then \( G \) contains an end-cycle.

**Proof:** Denote the set of the nonseparable components of \( G \) by \( N \). By definition of cacti, \( N=C \cup X \) where \( C \) is a set of cycles and \( X \) is a set of edges not on cycles. Let \( T=(N \cup S, E_T) \) be the superstructure of \( G \). Chose a vertex \( s_i \in S \) to be the root of \( T \) (see Figure 3). For any other vertex \( v \), let \( f(v) \) denote the neighbor of \( v \) which is on the unique path between \( s_i \) and \( v \). If \( u=f(v) \) then \( v \) is said to be a son of \( u \). The transitive closure of the son relation is the descendant relation. For each vertex \( v \) of \( T \), let \( A(v) \) denote the set of the neighbors of \( v \), that is, \( A(v)=\{u \mid (u,v) \in E_T\} \). Let \( c \in C \) be vertex
whose distance from \( s \) is maximal over all vertices of \( C \), and let \( s = f(c) \) (observe that \( s \in S \)). Let \( G' \) be the connected component in \( G \) induced by \( c \) and its descendants in \( T \). It is next shown that \( G' \) is an end-cycle in \( G \).

(Insert Figure 3 here)

(1) By the choice of \( c \) and the fact that \( G \) is trimmed, \( G' \) is a crown. Moreover, \( g(s) \) separates \( G' \) from \( G \).

(2) If \( (A(s) \cap C) = \{c\} \) then \( f(s) \in X \), so \( g(f(s)) \in E_o \). Let \( g(f(s)) = (v_1, v_2) \) where \( v_1 = g(s) \). Observe that \( v_2 \) separates \( G' \) from all other cycles in \( G \), for otherwise there would be a vertex \( c' \in C \), a descendant of \( s \), whose distance from the root is bigger than that of \( c \). Hence, \( G' \) is an end-cycle in \( G \).

(3) Assume that \( A(s) \) contains a vertex of \( C \) other than \( c \). By the choice of \( c \), no descendant of \( s \) is a vertex of \( C \), unless it belongs to \( A(s) \). Thus, no component in \( CC(g(s)) \), except perhaps for the one corresponding to \( g(f(s)) \), contains more than one cycle. We conclude that \( G' \) is an end-cycle. □

Lemma 1 can be used to convert a cactus into a trimmed cactus with no semi-forks, where all end-cycles are either of order 2 or of order 3. The rest of the edge-deletion rules are concerned with such end-cycles. The following lemma is proved using Proposition 1.

Lemma 3 [9] (Deletions due to end-cycles of order 2 or isolated cycles): Let \( G = (V_o, E_o) \) be a cactus. Let \( C \) be a subgraph of \( G \) which is either an isolated cycle or an end-cycle of order 2. Let \( v_1, v_2, ..., v_k \) be the vertices on \( C^o \) (starting from the anchor, if such exists). Then \( G' = (V_o, E'_o) \) where \( E'_o = E_o - \{(v_1, v_2)\} \) satisfies \( \pi(G') = \pi(G) \). □

Suppose that neither Lemma 1 nor Lemma 3 is applicable to a cactus \( G \). Then each end-cycle in \( G \) is of order 3. The following lemmas are concerned with such end-cycles.

Definition 4: Let \( G = (V_o, E_o) \) be a cactus. Let \( tr(v) = (v, v_1, ..., v_k) \) be the single trail starting at a vertex \( v \in V_o \). Then \( tr^{-1}(v) \) is the path \((v_k, ..., v_1, v)\). Let \( p_1 = (v_1, ..., v_k) \) and \( p_2 = (u_1, ..., u_l) \) be two paths in \( G \), where \((v_k, u_1) \in E_o \). Then \( p_1, p_2 \) is the path \((v_1, ..., v_k, u_1, ..., u_l)\).

Lemma 4 (Switching edges in a cover of an end-cycle of order 3): Let \( G = (V_o, E_o) \) be a cactus. Let \( C \subseteq G \) be an end-cycle of order 3 where \( v_1, v_2, v_3, ..., v_k \) are the vertices on \( C^o \), starting from the anchor. Let \( S_o \) be an optimal cover of \( G \) which employs \((v_1, v_2)\) but not \((v_1, v_k)\). Then there exists an optimal cover of \( G \), denoted by \( S \), which employs \((v_1, v_2)\) but not \((v_1, v_k)\).

Proof: Let \( p = p_1, p_2 \) be the path in \( S_o \) which employs \( e \), where \( p_1 \) may be empty (i.e. contain no vertices) and \( p_2 = (v_1, v_2, ..., v_n) \), \( n \geq 2 \). \( S \) is defined as follows. All paths in \( S_o \) that do not cover vertices in \( C \) are also in \( S \). Observe that since \( C \) contains \( k-1 \) vertices of degree 1 and \( p \) contains at most one of them, at least \( \lceil k/2 \rceil \) additional paths are
used by $S_G$ to cover the vertices in $C$ and in $p_1$. To prove the lemma, it suffices to show that $S_G$ uses exactly $\left\lfloor \frac{n}{2} \right\rfloor$ paths to cover those vertices, employing $(v_1,v_k)$ but not $(v_1,v_2)$. This is established by having $S_G$ cover those vertices using the paths $p_1, p_2, \ldots, p_{\left\lfloor \frac{n}{2} \right\rfloor}$ as follows (see Figure 4, where the bold edges are employed by $S_G$):

\begin{itemize}
  \item[(1)] $p_1 = p' \cdot (v_1) \cdot tr(v_k)$.
  \item[(2)] For $1 < i < \left\lfloor \frac{n}{2} \right\rfloor$, $p_i = tr^{-1}(v_{k-2i+3}) \cdot tr(v_{k-2i+2})$.
  \item[(3)] $p_{\left\lfloor \frac{n}{2} \right\rfloor} = tr^{-1}(v_3) \cdot tr(v_2)$ if $k$ is even, and $p_{\left\lfloor \frac{n}{2} \right\rfloor} = tr(v_2)$ if $k$ is odd. \qed
\end{itemize}

\textbf{Definition 5:} Let $G=(V_G, E_G)$ be a cactus. An \textit{anchor-trailed end-cycle} of $G$ is an end-cycle of order 3 where the anchor is the start-point of a trail.

\textbf{Lemma 5} (Deletions due to anchor-trailed end-cycles): Let $G=(V_G, E_G)$ be a cactus. Let $C \in G$ be a anchor-trailed end-cycle, where $v_1, v_2, \ldots, v_k$ are the vertices on $C$, starting from the anchor. Then $G'=(V_G, E'_G)$ where $E'_G = E_G - \{(v_1, v_2)\}$ satisfies $\pi(G') = \pi(G)$.

\textbf{Proof:} Clearly $\pi(G) \leq \pi(G')$. To prove the reverse inequality (hence equality), we show that every optimal cover of $G$ defines an equal-size cover of $G'$. Let $S_G$ be an optimal cover of $G$.

\begin{itemize}
  \item[(1)] If $(v_1,v_2)$ is not employed by $S_G$ then we are done, since $S_G$ is also a cover of $G'$.
  \item[(2)] If $(v_1,v_2)$ is employed by $S_G$ but $(v_1,v_k)$ is not, then by Lemma 4 there exists an equal-size cover of $G$ where $(v_1,v_2)$ is not employed but $(v_1,v_k)$ is. This cover is also a cover of $G'$.
  \item[(3)] Suppose that $(v_1,v_2)$ and $(v_1,v_k)$ are both employed by $S_G$. Let $u$ be the second vertex on a trail starting at $v_1$; observe that $(u,v_1)$ is not employed by $S_G$. By modifying $S_G$ to employ $(u,v_1)$ rather than $(v_1,v_2)$, one obtains, using Proposition 1, an equal-size cover of $G$ where $(v_1,v_2)$ is not employed. This cover is also a cover of $G'$.
\end{itemize}

\textbf{Definition 6:} Let $G=(V_G, E_G)$ be a cactus. A \textit{bridged end-cycle} of $G$ is an end-cycle of order 3 where the anchor is of degree 3.

\textbf{Lemma 6} (Deletions due to bridged end-cycles): Let $G=(V_G, E_G)$ be a cactus. Let $C \in G$ be a bridged end-cycle, where $v_1, v_2, \ldots, v_k$ are the vertices on $C$, starting from the anchor. Then $G'=(V_G, E'_G)$ where $E'_G = E_G - \{(v_2,v_3)\}$ satisfies $\pi(G') = \pi(G)$.

\textbf{Proof:} Clearly $\pi(G) \leq \pi(G')$. To prove the reverse inequality, we show that every optimal cover of $G$ defines an equal-size cover of $G'$. Let $S_G$ be an optimal cover of $G$. If $(v_2,v_3)$ is not employed by $S_G$ then we are done, since $S_G$ is also a cover of $G'$.
Suppose that \((V_1, V_2)\) is employed by \(S_o\). Let \(u\) be the second vertex on \(tr(v_2)\). Since \((v_2, v_3)\) is employed by \(S_o\), \((u, v_2)\) is not employed by \(S_o\). By modifying \(S_o\) to employ \((u, v_2)\) rather than \((v_2, v_3)\) one obtains, using Proposition 1, an equal-size cover of \(G\) where \((v_2, v_3)\) is not employed. This cover is also a cover of \(G'\).

(2) Suppose that \((v_1, v_2)\) is not employed by \(S_o\) but \((v_1, v_k)\) is. Then by Lemma 4, there exists an equal-size cover of \(G\) where \((v_1, v_k)\) is not employed but \((v_1, v_2)\) is, and the argument of (1) above applies.

(3) Suppose that neither \((v_1, v_2)\) nor \((v_1, v_k)\) is employed by \(S_o\). In this case, \(v_1\) is the end-vertex of some path \(p \in S_o\) and \((u, v_2)\) is employed by \(S_o\). By modifying \(S_o\) to employ \((v_1, v_2)\) rather than \((v_2, v_3)\) one obtains, using Proposition 1, an equal-size cover of \(G\) where \((v_2, v_3)\) is not employed. This cover is also a cover of \(G'\).

Definition 7: Let \(G=(V_o, E_o)\) be a cactus. An odd-trailed end-cycle of \(G\) is an end-cycle of order 3, \(C_o\), where the number of trails starting on \(C_o\) is odd. An even-trailed end-cycle of \(G\) is an end-cycle of order 3, \(C_e\), where the number of trails starting on \(C_e\) is even.

Lemma 7 (Deletions due to odd-trailed end-cycles): Let \(G=(V_o, E_o)\) be a cactus. Let \(C \subseteq G\) be an odd-trailed end-cycle where \(v_1, v_2, \ldots, v_k\) are the vertices on \(C_o\), starting from the anchor. Then \(G'=(V_o, E'_o)\) where \(E'_o=E_o-{(v_1, v_2)}\) satisfies \(\pi(G')=\pi(G)\).

Proof: Clearly \(\pi(G) \leq \pi(G')\). To prove the reverse inequality, we show that every optimal cover of \(G\) defines an equal-size cover of \(G'\). Let \(S_o\) be an optimal cover of \(G\).

(1) If \((v_1, v_2)\) is not employed by a path in \(S_o\) then we are done, since \(S_o\) is also a cover of \(G'\).

(2) If \((v_1, v_2)\) is employed by \(S_o\) but \((v_1, v_k)\) is not, then by Lemma 4 there exists an equal-size cover of \(G\) where \((v_1, v_2)\) is not employed but \((v_1, v_k)\) is. This latter cover is also a cover of \(G'\).

(3) Suppose that \((v_1, v_2)\) and \((v_1, v_k)\) are both employed by \(S_o\). Clearly, every path of \(S_o\) either covers only vertices in \(C\) or only vertices not in \(C\). In this case, an equal-size cover of \(G\), denoted by \(S_o\), can be constructed as follows. All paths in \(S_o\) that do not cover vertices in \(C\) are also in \(S_o\). Observe that at least \(\lceil \frac{k}{2} \rceil\) additional paths are used by \(S_o\) to cover the vertices in \(C\). To prove that \(S_o\) is an optimal cover of \(G'\), it suffices to show that it uses exactly \(\lceil \frac{k}{2} \rceil\) paths to cover those vertices, without employing \((v_1, v_2)\). This is established by having \(S_o\) cover those vertices using the paths \(p_1, p_2, \ldots, p_{\lceil \frac{k}{2} \rceil}\), where \(p_1=v_1 \cdot tr(v_k)\) and for \(1 \leq i \leq \lceil \frac{k}{2} \rceil\), \(p_i=tr^{-1}(v_{k-2i+2}) \cdot tr(v_{k-2i+3})\).

Definition 8: Let \(C=(V_c, E_c)\) be a crown where \(C^o\) contains an odd number of vertices \(v_1, v_2, \ldots, v_k\), each of which is of degree 3 except for \(v_1\). Then \(C^o\) is defined to be the graph induced on \(V_c-{v_1}\). Define \(\Delta(C)\) to be a cover of \(C\) using
\( \frac{k-1}{2} \) paths as follows:

1. \( p_1 = tr^{-1}(v_2) \cdot tr(v_k) \).

2. If \( k > 3 \), then for \( 1 \leq i \leq \frac{k-1}{2} \), \( p_i = tr^{-1}(v_{2i}) \cdot tr(v_{2i+1}) \).

Define \( \Lambda(C^v) \) to be a cover of \( C^v \) using \( \frac{k-1}{2} \) paths as follows: For \( 1 \leq i \leq \frac{k-1}{2} \), \( p_i = tr^{-1}(v_{2i}) \cdot tr(v_{2i+1}) \).

Lemma 8 (Covering even-trailed end-cycles using \( \Lambda \) and \( \Delta \)): Let \( G = (V_G, E_G) \) be a cactus. Let \( C \subseteq G \) be an even-trailed end-cycle where \( v_1, v_2, \ldots, v_k \) are the vertices on \( C^v \), starting from the anchor. Let \( S_G \) be an optimal cover of \( G \).

1. If \( S_G \) employs neither \((v_1, v_2)\) nor \((v_1, v_k)\), then \( S_G \) uses \( \Lambda(C^v) \) to cover \( C^v \).

2. If \( S_G \) employs both \((v_1, v_2)\) and \((v_1, v_k)\), then \( S_G \) uses \( \Delta(C) \) to cover \( C \).

Proof:

1. Let \( S \) be the set of paths used by \( S_G \) to cover \( C^v \). If \( S \neq \Lambda(C^v) \), then there exists a path \( p=(u_1, \ldots, u_k) \in S \) where at least one vertex from \( \{u_1, u_k\} \) is not the end-vertex of a trail. Since there are \( k-1 \) trails starting on \( C^v \), \( S - \{p\} \) must cover at least \( k-2 \) vertices of degree 1, using at least \( \left\lfloor \frac{k-2}{2} \right\rfloor = \frac{k-1}{2} \) paths. Thus, \( |S| = 1 + |S - \{p\}| = 1 + \frac{k-1}{2} = \frac{k+1}{2} \). Since \( |\Lambda(C^v)| = \frac{k-1}{2} \), a contradiction to the optimality of \( S_G \) arises.

2. The proof for this case is similar to that former case, and is omitted. \( \square \)

Lemma 9 (Deletions due to even-trailed end-cycles that share a vertex): Let \( G = (V_G, E_G) \) be a cactus. Let \( X = \{C_1, C_2, \ldots, C_a\} \) (\( a > 1 \)) be a set of even-trailed end-cycles in \( G \), all sharing the anchor \( v_1 \). Let \( v_1, v_2, \ldots, v_k \) be the vertices on \( C_1^v \), starting from the anchor. Then \( G' = (V_G, E_G') \) where \( E_G' = E_G - \{(v_1, v_2)\} \) satisfies \( \pi(G') = \pi(G) \).

Proof: Clearly \( \pi(G) \leq \pi(G') \). To prove the reverse inequality, we show that every optimal cover of \( G \) defines an equal-size cover of \( G' \). Let \( S_G \) be an optimal cover of \( G \).

1. If \((v_1, v_2)\) is not employed by a path in \( S_G \) then we are done, since \( S_G \) is also a cover of \( G' \).

2. If \((v_1, v_2)\) is employed by \( S_G \) but \((v_1, v_k)\) is not, then by Lemma 4 there exists an equal-size cover of \( G \) that employs \((v_1, v_k)\) but not \((v_1, v_2)\). This latter cover is also a cover of \( G' \).

3. Suppose that \((v_1, v_2)\) and \((v_1, v_k)\) are both employed by \( S_G \). Then in some other even-trailed end-cycle \( C_i \in X \), neither of the edges incident to \( v_1 \) is employed. By Lemma 8, \( S_G \) covers \( C \) and \( C_i^v \) using \( \Delta(C) \) and \( \Lambda(C_i^v) \), respectively. Observe that an equal-size cover of \( G \) is given by \( ((S_G - \Delta(C)) - \Lambda(C_i^v)) \cup \Lambda(C_i^v) \cup \Delta(C_i) \). This latter cover does not employ \((v_1, v_2)\), and is thus a cover of \( G' \). \( \square \)
3. A Recursive Decomposition Rule

In this section we consider end-cycles to which neither of the above edge-deletion rules is applicable. Let 
\( G=(V_\alpha, E_\alpha) \) be a cactus that properly contains a cycle. If \( G \) contains no semi-forks (hence, no forks), then by Lemma 2 \( G \) contains an end-cycle. Moreover, all end-cycles in \( G \) are either of order 2 or of order 3. Assume that no end-cycle of \( G \) is anchor-trailed, bridged, odd traile\( 33 \), or an even-trailed that shares its anchor with other even-trailed end-cycles. Then any 
end-cycle \( C \in G \) must be even-trailed where the degree of the anchor is 4. The recursive decomposition rule applies to such end-cycles.

Definition 9: Let \( G=(V_\alpha, E_\alpha) \) be a cactus. A final even-trailed end-cycle of \( G \) is an even-trailed end-cycle where the anchor is of degree 4. Let \( C=(V_e, E_e) \) be a final even-trailed end-cycle in \( G \), where \( v_1, v_2, \ldots, v_k \) are the vertices on \( C^o \) starting from the anchor. Let where \( u \) and \( w \) are the vertices adjacent to \( v_1 \) that are not on \( C^o \). Then \( GIC \) is defined to be the 
graph \( GIC=(V_\alpha-V_e, E_\alpha-E_e) \) where \( E_\alpha-E_e \cup \{(u, w)\} \) (see Figure 5).

Lemma 10 (Recursive decomposition rule for final even-trailed end-cycles, part I): Let \( G=(V_\alpha, E_\alpha) \) be a cactus. Let \( C \in G \) be a final even-trailed end-cycle where \( v_1, v_2, \ldots, v_k \) are the vertices on \( C^o \) starting from the anchor. Then 
\[ \pi(GIC) \leq \pi(G) - \frac{k-1}{2} \]

Proof: Let \( S_\alpha \) be an optimal cover of \( G \). It is proved that \( S_\alpha \) defines a cover \( S_{\alpha C} \) of \( GIC \) such that 
\[ |S_{\alpha C}| = |S_\alpha| - \frac{k-1}{2}. \]

In the sequel, let \( u \) and \( w \) be the vertices adjacent to \( v_1 \) that are not on \( C^o \).

1) Suppose that neither \( (u, v_1) \) nor \( (v_1, w) \) is employed by \( S_\alpha \). In this case, it is easily verified that \( S_\alpha \) uses \( \Delta(C) \) to cover 
\( C \). Observe that for each edge \( e \in E_\alpha \), if \( e \) is employed by \( S_\alpha-\Delta(C) \) then \( e \) is an edge in \( GIC \). Thus, \( S_{\alpha C}=S_\alpha-\Delta(C) \)
is a cover of \( GIC \). Its size is given by 
\[ |S_{\alpha C}| = |S_\alpha| \cdot \frac{k-1}{2}. \]

2) Suppose that exactly one edge from \( \{(u, v_1), (v_1, w)\} \), say \( e=(u, v_1) \), is employed by \( S_\alpha \). Let \( p=p_1-(u, v_1)-p_2 \) be the 
path in \( S_\alpha \) which employs \( e \), where \( p_1 \) and \( p_2 \) may be empty. Using Proposition 1 and the fact that \( C \) is even-trailed, 
the reader can verify that an equal-size cover of \( G \) which employs neither \( (u, v_1) \) nor \( (v_1, w) \) but both \( (v_1, v_2) \) and 
\( (v_1, v_k) \) can be obtained by replacing \( p \) by \( p_1-(u) \) and using \( \Delta(C) \) to cover \( C \). From here, the argument of (1) above applies.

3) Suppose that both \( (u, v_1) \) and \( (v_1, w) \) are employed by \( S_\alpha \). Then neither \( (v_1, v_2) \) nor \( (v_1, v_k) \) is employed by \( S_\alpha \). By 
Lemma 8, \( S_\alpha \) uses \( \Delta(C^o) \) to cover \( C \). Let \( p \in S_\alpha \) be the path employing \( (u, v_1) \) and \( (v_1, w) \), and let \( p' \) be the path 
obtained by deleting \( v_1 \) from \( p \). Then 
\[ S_{\alpha C}=((S_\alpha-\Delta(C^o))-(p_1)) \cup \{p'\} \]
is a cover of \( GIC \), and its size is
Lemma 11 (Recursive decomposition rule for final even-trailed end-cycles, part II): Let \( G=(V_o,E_o) \) be a cactus. Let \( C \in G \) be a final even-trailed end-cycle where \( v_1, v_2, \ldots, v_k \) are the vertices on \( C^o \), starting from the anchor. Let \( u \) and \( w \) be the vertices adjacent to \( v_1 \) that are not on \( C \), and let \( S_{alc} \) be an optimal cover of \( G \mid C \).

1) Suppose that some path \( p \in S_{alc} \) employs \((u,w)\). Let \( p' \) be the path obtained from \( p \) by inserting \( v_1 \) between \( u \) and \( w \). Then \( S_o=(S_{alc} \setminus \{p\}) \cup p' \cup \Lambda(C^o) \) is an optimal cover of \( G \).

2) If \( S_{alc} \) does not employ \((u,w)\), then \( S_o=S_{alc} \cup \Delta(C) \) is an optimal cover of \( G \).

Proof: In both cases, \(|S_o|=|S_a| \leq \frac{k-1}{2}\). Combining this fact with Lemma 10, we conclude that \( S_o \) is optimal. \( \square \)

4. The Algorithm

In this section we present an algorithm for optimal covering of cacti. A first version of the algorithm, called Algorithm A, is given below. The purpose of this version is to demonstrate the algorithmic use of the edge-deletion rules and the recursive decomposition rule. In doing so, we focus on simplicity rather than efficiency. An efficient (and more complicated) algorithm is described later.

Informally, Algorithm A runs as follows. The edge-deletion rules are repeatedly applied to delete edges from the input cactus, \( G \). When neither of the edge-deletion rules is applicable any more, the recursive decomposition rule is invoked, and the algorithm is recursively applied to the resulting graph. Eventually, \( G \) reduces to a set of paths which constitutes an optimal cover of \( G \).

We next review some definitions that were given in the previous sections, to be used by the algorithm. Let \( G=(V_o,E_o) \) be a cactus. A vertex \( v \in V_o \) is a fork if \( \deg(v) \geq 3 \) and at least two trails start at \( v \). A vertex \( v_1 \in V_o \) is a semi-fork if \( \deg(v_1) \geq 3 \), exactly one trail \((v_1,v_2,\ldots)\) starts at \( v_1 \), and \( v_1 \) is adjacent to a vertex \( w \) where \( \deg(w)=2 \) (\( w \neq v_2 \)).

A anchor-trailed end-cycle is an end-cycle of order 3 where the anchor is the start-point of a trail. A bridged end-cycle is an end-cycle of order 3 where the anchor is of degree 3. An odd-trailed end-cycle is an end-cycle of order 3, \( C \), where the number of trails starting on \( C^o \) is odd. An even-trailed end-cycle is an end-cycle of order 3, \( C \), where the number of trails starting on \( C^o \) is even. A final even-trailed end-cycle is an even-trailed end-cycle where the anchor is of degree 4.

We are now able to present our covering algorithm.

Algorithm A

Input: A cactus \( G=(V_o,E_o) \).

Output: A set of paths \( S_o \), comprising an optimal cover of \( G \).

Procedure used:

Procedure Transfer-Paths;
do
   Add the isolated paths in $G=(V_o, E_o)$ to $S_o$.
   $V_o \leftarrow V_o - \{ v \in V_o \mid \text{some path in } S_o \text{ covers } v \}$.
   $E_o \leftarrow E_o - \{ e \in E_o \mid \text{some path in } S_o \text{ employs } e \}$.
   od

Method:
1. Initialize $S_o \leftarrow \emptyset$.
2. Transfer-Paths.
3. While $G=(V_o, E_o)$ contains forks,
   3.1 Choose a fork $v$.
   3.2 Apply Lemma 1 to $v$.
   3.3 Transfer-Paths.
   (* Comment: At this point, $G$ is a union of trimmed cacti. *)
4. If $G=(V_o, E_o)$ contains a semi-fork $v$, then
   4.1 Apply Lemma 1 to $v$.
   4.2 Go to step 3.
   (* Comment: At this point, all end-cycles in $G$ are either of order 2 or of order 3. *)
5. If $G=(V_o, E_o)$ contains a subgraph $C$ which is either an isolated cycle or an end-cycle of order 2, then
   5.1 Apply Lemma 3 to $C$.
   5.2 Transfer-Paths.
   5.3 Go to step 3.
   (* Comment: At this point, all end-cycles in $G$ are of order 3. *)
6. If $G=(V_o, E_o)$ contains an anchor-trailed end-cycle, $C$, then
   6.1 Apply Lemma 5 to $C$.
   6.2 Go to step 3.
7. If $G=(V_o, E_o)$ contains a bridged end-cycle, $C$, then
   7.1 Apply Lemma 6 to $C$.
   7.2 Go to step 3.
8. If $G=(V_o, E_o)$ contains an odd-trailed end-cycle, $C$, then
   8.1 Apply Lemma 7 to $C$.
   8.2 Go to step 3.
   (* Comment: At this point, all end-cycles in $G$ are even-trailed end-cycles. *)
9. If $G=(V_o, E_o)$ contains two or more even-trailed end-cycles that share a vertex $v$, then
   9.1 Choose an even-trailed end-cycle $C$ from these sharing $v$.
   9.2 Apply Lemma 9 to $C$.
   9.3 Go to step 3.
   (* Comment: At this point, all end-cycles in $G$ are final even-trailed end-cycles. *)
10. If $G=(V_o, E_o)$ contain a final even-trailed end-cycle, $C$, then
   10.1 Let $v$ be the anchor of $C$. Let $u$ and $w$ be the vertices adjacent to $v$ that are not on $C^*$.
   10.2 Recursively apply the algorithm to $G\setminus C$, resulting in an optimal cover $S_{ac}$.
   10.3 If $(u, w)$ is employed by a path $p \in S_{ac}$, then
   10.3.1 Let $p'$ the path obtained from $p$ by inserting $v$ between $u$ and $w$.
   10.3.2 $S_o \leftarrow S_o \cup (S_{ac} - \{p\}) \cup p' \cup A(C^*)$
10.4 If \( p \) is not employed by \( S_{ac} \), then \( S_{o} \leftarrow S_{o} \cup \left( S_{ac} \cup \Delta(C) \right) \).

11 Stop.

**Theorem 1 (Correctness of Algorithm A):** Given a cactus \( G=(V_o, E_o) \), Algorithm A produces an optimal path cover of \( G \).

**Proof:** Whenever the algorithm returns to step 3, the size of \( E_o \) is strictly smaller than it was in the previous execution of step 3. Thus, the algorithm eventually terminates, since none of the conditions tested in steps 3-10 holds when \( E_o = \emptyset \).

Upon termination, \( G \) contains no forks, semi-forks, isolated cycles, or end-cycles. Hence, by Lemma 2, \( G \) contains no non-isolated cycles. Also, \( G \) contains no isolated paths upon termination, for such paths, which are generated only by applying Lemmas 1 and 3, are immediately transferred to \( G \). It follows that upon termination \( V_o = \emptyset \), so \( S_o \) is a cover of \( G \).

The algorithm deletes edges from \( G \) only by applying the edge-deletion rules. The decomposition rule ensures that the construction of an optimal cover, upon return from each recursive invocation of the algorithm, is properly done. We conclude that when the algorithm terminates, \( |S_o| = \pi(G) \). \( \square \)

Using the fact that the number of cycles in a cactus \( G=(V_o, E_o) \) is \( O(|E_o|) \), the reader can verify that Algorithm A can be implemented in \( O(|E_o|^2) = O(|V_o|^2) \) time. However, a better bound is in fact achievable by Algorithm B below. Algorithm B is based on the DFS (depth first search) algorithm ([7], see also [4]), with which we assume the reader is familiar.

**Definition 10:** An EDFS is a DFS extended to identify forks/semi-forks upon backtracking from such. Recall that DFS generates a directed tree, where edges not in the tree are called back-edges. Assume that an EDFS is applied to a cactus \( G=(V_o, E_o) \), and that \( e \in E_o \) is a back-edge of the EDFS tree. Then \( C(e) \) is defined to be the unique cycle in \( G \) which contains \( e \). The source of a cycle \( C \) with respect to the EDFS is the first vertex on \( C \) which was discovered by the EDFS. Note that if \( v \) is a source of a cycle, \( C \), then there is a back-edge \( (u \rightarrow v) \) on \( C \) entering \( v \). Let \( G=(V_o, E_o) \) be a graph, and let \( v \in V_o \) be a separation vertex of \( G \). Assume that one of the connected components which \( v \) separates from \( G \) is a tree, \( T \). An elimination of \( T \) from \( G \) is an EDFS traversal of \( T \), starting from \( v \), where Lemma 1 is applied to each fork \( u \in V_o \setminus \{v\} \) upon backtracking from \( u \).

Algorithm B, whose time complexity is an \( O(|V_o|) \), is outlined below. It is based on an EDFS traversal of the input cactus. In the course of the EDFS, the edge-deletion rules, as well as the recursive decomposition rule, are applied to \( G \), resulting in a properly smaller graph. Specifically, these rules are applied whenever the algorithm backtracks from a fork that is not on a cycle, or from a source of a cycle. The isolated paths created by applying the edge-deletion rules are transferred to set \( S_o \), which eventually constitutes an optimal cover of \( G \). Whenever a final even-trailer end-cycle, \( C \), is detected, the two
vertices adjacent to the anchor of $C$ are connected to form $G\mid C$; $C$ is then pushed onto a stack, to be covered when $G\mid C$ is fully covered.

Algorithm B

Initialize $S_\sigma \leftarrow \emptyset$. Starting from an arbitrary vertex, traverse $G$ using EDFS. Immediately before backtracking from a vertex $v$, invoke procedure $\text{Backtrack-From}(v)$, described below.

Procedure $\text{Backtrack-From}(v)$

do
1. Record the father of $v$.
2. If $v$ is not on a cycle and is a fork, apply Lemma 1 to $v$.
3. If $v$ is the source of a cycle, perform the following:

3.1. Let $B_v$ be the set of EDFS back-edges entering $v$. Temporarily suspend the EDFS, and for each $e = (u \rightarrow v) \in B_v$, re-traverse the remaining edges of $C(e)$ - backtracking the EDFS tree-edges, starting from $u$. Stop the re-traversal of $C(e)$ upon discovering a fork or a semi-fork. If none exists - count the number of trails starting on $C(e)$.

3.2. Let $S = \{ e \in B_v \mid C(e) \text{ contains a fork or a semi-fork} \}$. For each $e \in S$, apply Lemma 1 to a fork or a semi-fork on $C(e)$, transferring the isolated paths thus created to $S_\sigma$.

3.3. The remaining edges of the cycles of $S$ induce a tree which is separated from $G$ by $v$. Eliminate this tree from $G$, transferring the isolated paths thus created to $S_\sigma$.

3.4. While $B_v$ contains an edge $e$ such that no edge on $C(e)$ was deleted, perform the following:

3.4.1. If $C(e)$ is an isolated cycle or is contained in an end-cycle, $C$, that satisfies the requirements of some edge-deletion lemma, perform the following:

Apply that lemma to $C$. The remaining edges of $C$ now induce a tree which is separated from $G$ by $v$. Eliminate this tree from $G$, transferring the isolated paths thus created to $S_\sigma$.

3.4.2. If $C(e)$ underlies a final even-trailed end-cycle, $C$, perform the following:

Let $u$ and $w$ be the vertices adjacent to $v$ that are not on $C$. Delete all the vertices of $C$ from $G$ and connect $u$ and $w$ to form $G\mid C$. Push $C$ onto the stack of the vet-uncovered
final even-trailed end-cycles.

\[
\text{od}
\]

On completion of Backtrack-From(\(v\)), resume the EDFS from the father of \(v\). When the EDFS is done, pop and cover the even-trailed end-cycles stored in the stack, using the recursive decomposition rule.

Theorem 2 (Correctness of Algorithm B): Given a cactus \(G=(V_G, E_G)\), Algorithm B produces an optimal cover of \(G\) in \(O(|V_G|)\) time.

Proof (Outline): The theorem can be established using the following claims, which, in turn, can be verified by standard techniques for proving the correctness of DFS-based algorithms. In the sequel, we say that \(u\) is a descendant of \(v\) at a given time within the execution of the algorithm, if, at that time, \(u\) is reachable from \(v\) by a sequence of tree edges \((v \rightarrow x_1), (x_1 \rightarrow x_2), \ldots, (x_{n-1} \rightarrow x_n), (x_n \rightarrow u)\).

1. Upon invoking Backtrack-From(\(v\)), \(v\) is on a cycle iff there is a back-edge entering \(v\) or lowpoint(\(v\)) \(\leq k(\(v\))\) (see [4]). Hence, checking if \(v\) is on a cycle can be done in constant time.

2. Upon invoking Backtrack-From(\(v\)), if \(v\) is not on a cycle then no descendant of \(v\) is a fork or on a cycle.

3. Upon invoking Backtrack-From(\(v\)), if \(v\) is the source of cycles \(C_1, C_2, \ldots, C_n\), then no descendant of \(v\), except perhaps for those on some \(C_i\) (1 \(\leq i \leq n\)), is a fork or on a cycle. Hence, for 1 \(\leq i \leq n\), \(C_i\) is either an end-cycle or contains a fork.

4. If \(v\) is the source of cycles \(C_1, C_2, \ldots, C_n\) upon invoking Backtrack-From(\(v\)), then on completion of Backtrack-From(\(v\)), no descendant of \(v\) is a fork or on a cycle.

5. Each edge \(e \in E_G\) is scanned a constant number of times (twice by the EDFS, at most once by the re-traversal, and at most twice by either the elimination process or the covering of the final even-trailed end-cycles). □

References


Figure 1. A Cactus (reproduced from [1])

Figure 2. End-cycles
Figure 3: The superstructure of Lemma 2
Figure 4. The new cover of Lemma 4
Figure 5. Definition of GIC