The Algorithmic Complexity of Perfect Neighborhoods in Graphs

Sandra Hedetniemi
Stephen T. Hedetniemi
Department of Computer Science
Clemson University
Clemson, SC 29634-1906

Michael A. Henning *
Department of Mathematics
University of Natal
Private Bag X01
Pietermaritzburg, 3209 South Africa

ABSTRACT. Let G be a graph and let S be a subset of vertices of G. A vertex v of G is called perfect with respect to S if $|N[v] \cap S| = 1$ where $N[v]$ denotes the closed neighborhood of v. The set S is defined to be a perfect neighborhood set of G if every vertex of G is perfect or adjacent with a perfect vertex. The perfect neighborhood number $\theta(G)$ of G is defined to be the minimum cardinality among all perfect neighborhood sets of G. In this paper, we present a variety of algorithmic results on the complexity of perfect neighborhood sets in graphs.

1 Introduction

Let $G = (V, E)$ be a graph with vertex set V and edge set E, and let v be a vertex in V. The open neighborhood of $v \in V$ is $N(v) = \{u \in V \mid uv \in E\}$ and the closed neighborhood of v is $N[v] = \{v\} \cup N(v)$. For a set S of vertices, we define the open neighborhood $N(S) = \cup_{v \in S} N(v)$, and the closed neighborhood $N[S] = N(S) \cup S$.

*Research supported in part by the University of Natal and the South African Foundation for Research Development.

JCMCC 25 (1997), pp. 183-192

The concept of perfect neighborhood sets in graphs was introduced and studied in [2]. Let S be a subset of vertices of G. A vertex v of G is called *perfect* with respect to S if $|N[v] \cap S| = 1$. The set S is defined to be a *perfect neighborhood* set of G if every vertex of G is perfect or adjacent with a perfect vertex. The *(lower) perfect neighborhood number* $\theta(G)$ of G is defined to be the minimum cardinality among all perfect neighborhood sets of G. We will refer to a perfect neighborhood set of cardinality $\theta(G)$ as a θ-set of G.

Perfect neighborhood sets may be defined in terms of functions. Let $f: V \to \{0, 1\}$ be a function which assigns to each vertex of G an element of the set $\{0, 1\}$. For $S \subseteq V$, we define $f(S) = \sum_{v \in S} f(v)$, and we define the *weight* of f to be $w(f) = \sum_{v \in V} f(v) = f(V)$. For a vertex v in V, we denote $f(N[v])$ by $f[v]$ for notational convenience. Corresponding to each perfect neighborhood set S of G, we associate the function, namely the characteristic function f of S defined by $f(v) = 1$ if $v \in S$ and $f(v) = 0$ otherwise. Such a function we call a *perfect neighborhood function* of G. Hence a function $f: V \to \{0, 1\}$ is defined to be a perfect neighborhood function of a graph G if for every $u \in V$, there exists a $v \in N[u]$ satisfying $f[v] = 1$. The perfect neighborhood number of a graph G can be defined as the minimum weight of a perfect neighborhood function of G.

In this paper we present a variety of algorithmic results. We show that the decision problem corresponding to the problem of computing θ is NP-complete even when restricted to bipartite graphs or chordal graphs. Upper bounds on $\theta(G)$ are presented for connected graphs G. We also present a linear time algorithm for finding a perfect neighborhood set in an arbitrary tree.

2 Complexity results for θ

From a computational point of view the problem of finding $\theta(G)$ appears to be very difficult. In this section we show that the decision problem

PERFECT NEIGHBORHOOD SET (PNS)

INSTANCE: A graph $G = (V, E)$ and a positive integer $m \leq |V|$.

QUESTION: Is there a perfect neighborhood set of cardinality m?

is NP-complete, even when restricted to bipartite and chordal graphs, by describing polynomial transformations from the following well-known NP-complete problem:

EXACT COVER BY 3-SETS (X3C)

INSTANCE: A finite set X with $|X| = 3q$ and a collection C of 3-element subsets of X.

QUESTION: Does C contain an exact cover for X, that is, a subcollection $C' \subseteq C$ such that every element of X occurs in exactly one member of C'.

Theorem 1 *PNS is NP-complete, even for bipartite graphs.*

Proof. It is clear that PNS is in NP. To show that PNS is an NP-complete problem, we will establish a polynomial transformation from **X3C**. Let $X = \{x_1, \ldots, x_{3q}\}$ and $\mathcal{C} = \{C_1, \ldots, C_m\}$ be an arbitrary instance of **X3C**. We will construct a bipartite graph G such that this instance of **X3C** will have an exact three cover if and only if G has a perfect neighborhood set of cardinality m.

The graph G is constructed as follows. Corresponding to each variable $x_i \in X$, we associate the path x_i, y_i on two vertices. Corresponding to each set C_j, we associate the graph F_j which consists of the path a_j, b_j, c_j on three vertices. The construction of the bipartite graph G is completed by adding the edges $\{x_i c_j \mid x_i \in C_j\}$. It is easy to see that the construction of the graph G can be accomplished in polynomial time. Let $X = \{x_1, x_2, \ldots, x_{3q}\}$, $Y = \{y_1, y_2, \ldots, y_{3q}\}$, and $C = \{c_1, \ldots, c_m\}$. We show that $\mathcal{C}$ has an exact 3-cover if and only if G has a perfect neighborhood set of cardinality m.

Suppose $\mathcal{C}'$ is an exact 3-cover for X. Then $|\mathcal{C}'| = q$. Let $S = \{c_j \mid C_j \in \mathcal{C}'\} \cup \{b_j \mid C_j \in \mathcal{C} - \mathcal{C}'\}$. Then S is a perfect neighborhood set of G of cardinality m. Suppose, conversely, that S is a perfect neighborhood set of cardinality m. If S contains no vertex of F_j for some j, $1 \le j \le m$, then the vertex a_j is neither perfect nor adjacent with a perfect vertex with respect to S, contradicting the fact that S is a perfect neighborhood set of G. Thus, $|V(F_j) \cap S| \ge 1$ for $j = 1, \ldots, m$. Consequently, $|S \cap V(F_j)| = 1$ for $j = 1, \ldots, m$ and $S \cap (X \cup Y) = \emptyset$. Since S contains no vertex from $X \cup Y$, no vertex of Y is perfect with respect to S. Hence each vertex of Y must be adjacent with a perfect vertex. Thus each vertex of X must be perfect, and is therefore adjacent with a unique vertex of S. Letting $S' = S \cap C$, each vertex of X is therefore adjacent with exactly one vertex of S'. Hence there are precisely $3q$ edges joining X and S'. However, since each vertex of S' is adjacent with three vertices of X, there are $3|S'|$ edges joining X and S', so $|S'| = q$. Consequently, $\mathcal{C}' = \{C_j \mid c_j \in S\}$ is an exact 3-cover for X.

Theorem 2 *PNS is NP-complete, even for chordal graphs.*

Proof. It is clear that PNS is in NP. To show that PNS is an NP-complete problem, we will establish a polynomial transformation from **X3C**. Let $X = \{x_1, \ldots, x_{3q}\}$ and $\mathcal{C} = \{C_1, \ldots, C_m\}$ be an arbitrary instance of **X3C**. We will construct a chordal graph H such that this instance of **X3C** will have an exact three cover if and only if H has a perfect neighborhood set of cardinality m.

Let H be obtained from the graph G constructed in the proof of Theorem 1 by adding an edge between every two vertices of C so that the c_j's induce a clique; that is, $\langle\{c_1,\ldots,c_m\}\rangle \cong K_m$. It is easy to see that the construction of the graph H can be accomplished in polynomial time. Proceeding now as in the proof of Theorem 1, we may show that C has an exact 3-cover if and only if H has a perfect neighborhood set of cardinality m.

3 Bounds on θ

Since the problem of computing $\theta(G)$ appears to be a difficult one, it is desirable to find good upper bounds on this parameter. The concept of perfect neighborhood sets is in a some sense related to dominating sets. For a graph $G = (V, E)$, a subset S of vertices of G is defined to be a *dominating set* of G if for every vertex v in V, $|N[v] \cap S| \geq 1$. Equivalently, S is a dominating set of G if each $v \in V$ is either in S or adjacent to a vertex of S. (That is, $N[S] = V$.) The *domination number* of a graph G, denoted $\gamma(G)$, is the minimum cardinality of a dominating set in G. In [2], the following result is established.

Lemma 1 *For any minimal dominating set D of a graph G, there exists a perfect neighborhood set of G of cardinality $|D|$.*

Corollary 1 *For every graph G, $\theta(G) \leq \gamma(G)$.*

However, it is well known (see [3]) that $\gamma(G) \leq n/2$ for all connected graphs G of order $n \geq 2$. Thus we have the following result.

Proposition 1 *If G is a connected graph of order $n \geq 2$, then $\theta(G) \leq n/2$.*

That the upper bound in Proposition 1 is in a sense best possible, may be seen as follows. For $m \geq 2$ an integer, let T be the tree obtained from a star $K_{1,m}$ by subdividing each edge once. Let $T_1, T_2, \ldots, T_m$ be m disjoint copies of T, and let v_i denote the central vertex of T_i for $i = 1, 2, \ldots, m$. Finally, let G_m be the graph obtained from the disjoint union $\cup_{i=1}^m T_i$ of $T_1, T_2, \ldots, T_m$ by adding a new vertex v_{m+1} and the edges v_iv_j for $1 \leq i < j \leq m+1$. The graph G_m is shown in Figure 1. We show that $\theta(G_m) = m^2 - m + 1$. Let S be a perfect neighborhood set of G_m. Then S contains at most one of $v_1, v_2, \ldots, v_{m+1}$. If $v_i \notin S$, then $|S \cap V(T_i)| \geq m$ since every perfect neighborhood set is 2-dominating. Thus, $\theta(G_m) \geq 1 + (m-1)m = m^2 - m + 1$. However, there exist perfect neighborhood sets of G_m of cardinality $m^2 - m + 1$ as illustrated by the set of darkened vertices in Figure 1, so $\theta(G_m) \leq m^2 - m + 1$. Thus, $\theta(G_m) = m^2 - m + 1$. Hence G_m is a connected graph of order $n = 2m^2 + m + 1$ satisfying

$$\frac{\theta(G_m)}{n} = \frac{m^2 - m + 1}{2m^2 + m + 1} = \frac{1 - 1/m + 1/m^2}{2 + 1/m + 1/m^2}.$$

Hence, $\theta(G_m)/n \to 1/2$ as $m \to \infty$.

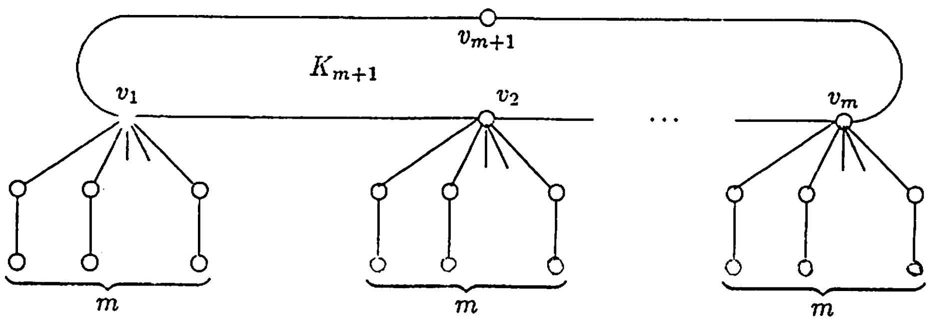

Figure 1. The graph G_m. (The darkened vertices form a perfect neighborhood set in G_m.)

If G is a connected graph with minimum degree at least 2, then the upper bound in Proposition 1 can be improved. A *packing* of a graph is a set of vertices whose elements are pairwise at distance at least 3 apart in G. In [2] the following result is established.

Lemma 2 *Every maximal packing S of a graph G is a perfect neighborhood set of G.*

Using Lemma 2, we may improve the upper bound in Proposition 1 when G is a connected graph with minimum degree at least 2.

Proposition 2 *Let $G = (V, E)$ be a connected graph of order $n \geq 3$ with minimum degree at least 2. Then $\theta(G) \leq n/3$.*

Proof. Let $S = \{v_1, v_2, \ldots, v_k\}$ be a maximal packing of G. Then $N[v_i] \cap N[v_j] = \emptyset$ for $1 \leq i < j \leq k$. Thus $|\cup_{i=1}^{k} N[v_i]| = \sum_{i=1}^{k} deg\, v_i + 1 \geq 3k = 3|S|$. On the other hand, $\cup_{i=1}^{k} N[v_i] \subseteq V$, so $|\cup_{i=1}^{k} N[v_i]| \leq n$. Thus, $|S| \leq n/3$. The result now follows from Lemma 2.

That the upper bound in Proposition 2 is in a sense best possible, may be seen as follows. Let F_m be the graph obtained from two disjoint copies of the graph G_m (contructed in the paragraph following Proposition 1) by identifying the corresponding end-vertices. (Equivalently, F_m is obtained from the disjoint union of two complete graphs K_{m+1} on $m+1$ vertices with the vertices from the one complete graph named $v_1, v_2, \ldots, v_{m+1}$ and from the other named $v'_1, v'_2, \ldots, v'_{m+1}$ by joining v_i and v'_i with m internally disjoint paths each of length 4 for $i = 1, 2, \ldots, m$.) Then F_m is a connected graph of order $n = m(3m+2)+2 = 3m^2+2m+2$ with $\theta(F_m) = 2+(m-2)m = m^2-2m+2$. Thus F_m satisfies

$$\frac{\theta(F_m)}{n} = \frac{m^2-2m+2}{3m^2+2m+2} = \frac{1-2/m+2/m^2}{3+2/m+2/m^2}.$$

Hence, $\theta(F_m)/n \to 1/3$ as $m \to \infty$.

4 A linear algorithm for computing $\theta(T)$ for a tree T

In this section, we present a linear algorithm for computing the value of $\theta(T)$ for any tree T. We construct a dynamic style algorithm using the methodology of Wimer (see [5]). We characterize the possible classes tree-subset pairs (T, S), give a multiplication table for these classes, describe the recursion relations (and the associated equations) and then give a complete algorithm. We make use of the well-known fact that the class of (rooted) trees can be constructed recursively from copies of the single vertex K_1, using only one rule of composition, which combines two trees (T_1, r_1) and (T_2, r_2) by adding an edge between r_1 and r_2 and calling r_1 the root of the resulting larger tree. We denote this composition of T_1 and T_2 by $T_1 \circ T_2$.

For notational convenience, we will denote a 'perfect neighborhood set' simply by a 'pn-set'. We begin by defining the collection of possible tree-subset pairs TS as the set of all ordered pairs (T, S) which satisfy the following three properties:

- T is a rooted tree with root r.
- If $r \in N[S]$, then every vertex of T, except possibly for r, is perfect or adjacent with a perfect vertex with respect to the set S.
- If $r \notin N[S]$, then every vertex of T, except for r and possibly for some neighbors of r, is perfect or adjacent with a perfect vertex with respect to the set S.

Then any such (tree-set) pair (T, S) can be classified into one of the following fourteen classes (unless otherwise stated, perfect vertices are with respect to the set S):

[1] $= \{(T, S) \mid r \in S$, S is a pn-set of T, r is perfect, r is adjacent with a perfect vertex, r has a neighbor that is not perfect and has r as its unique perfect neighbor$\}$,

[2] $= \{(T, S) \mid r \in S$, S is a pn-set of T, r is perfect, r is adjacent with a perfect vertex, every neighbor of r is perfect or is adjacent with a perfect vertex different from $r\}$,

[3] $= \{(T, S) \mid r \in S$, S is a pn-set of T, r is perfect, r is not adjacent with a perfect vertex, r is the unique perfect neighbor for one of its neighbors$\}$,

[4] $= \{(T, S) \mid r \in S$, S is a pn-set of T, r is perfect, r is not adjacent with a perfect vertex, r is not the unique perfect neighbor of any of its neighbors$\}$,

[5] $= \{(T, S) \mid r \in S$, S is a pn-set of T, r is not perfect$\}$,

$[6] = \{(T,S) \mid r \in S$, S is not a pn-set set of $T\}$,

$[7] = \{(T,S) \mid r \notin S$, S is a pn-set of T, r is perfect, r is adjacent with a perfect vertex, r has a neighbor that is not perfect and has r as its unique perfect neighbor$\}$,

$[8] = \{(T,S) \mid r \notin S$, S is a pn-set of T, r is perfect, r is adjacent with a perfect vertex, every neighbor of r is perfect or is adjacent with a perfect vertex different from $r\}$,

$[9] = \{(T,S) \mid r \notin S$, S is a pn-set of T, r is perfect, r is not adjacent with a perfect vertex, r is the unique perfect neighbor for one of its neighbors$\}$,

$[10] = \{(T,S) \mid r \notin S$, S is a pn-set of T, r is perfect, r is not adjacent with a perfect vertex, r is not the unique perfect neighbor of any of its neighbors$\}$,

$[11] = \{(T,S) \mid r \notin S$, S is a pn-set of T, r is not perfect$\}$,

$[12] = \{(T,S) \mid r \notin N[S]$, S is not a pn-set of T, every neighbor of r is adjacent with a perfect vertex $\}$,

$[13] = \{(T,S) \mid r \notin N[S]$, S is not a pn-set of T, r has a neighbor that is not adjacent with a perfect vertex $\}$,

$[14] = \{(T,S) \mid r \notin S$, S is not a pn-set of T, $|N(r) \cap S| \geq 2\}$,

These subclasses will be used in the design of the algorithm. For any tree-set pairs (T_1, S_1) and (T_2, S_2), we will denote the pair $(T_1 \circ T_2\ ,\ S_1 \circ S_2)$ with $(T_1, S_1) \circ (T_2, S_2)$. We now consider the effect of composing a tree T_1 having a set S_1 which is of class $[i]$ with a tree T_2 having a set which is of class $[j]$ for every possible combination of classes $1 \leq i < j \leq 14$. That is, we must describe the appropriate class of the combined set $S_1 \cup S_2$ in the composed tree $T = T_1 \circ T_2$. This is described in the multiplication table for these classes shown in Figure 2. The symbol $\times$ indicates that the resulting tree-set pair, though defined, is not a member of TS; that is, no set S can ever decompose into two subsets S_1 and S_2 of the classes indicated.

∘	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]
[1]	×	×	×	×	×	×	×	[1]	×	[1]	[1]	[1]	[1]	[1]
[2]	×	[5]	×	×	[5]	×	×	[2]	×	[1]	[2]	[2]	[2]	[1]
[3]	×	×	×	×	×	×	×	[3]	×	×	[3]	[1]	[1]	[3]
[4]	×	[6]	×	×	[6]	×	×	[4]	×	[3]	[4]	[2]	[2]	[3]
[5]	×	[5]	×	×	[5]	×	×	[5]	×	×	[5]	[5]	[5]	×
[6]	×	[6]	×	×	[6]	×	×	[6]	×	×	[6]	[5]	[6]	×
[7]	×	×	×	×	×	×	[7]	[7]	[7]	[7]	[7]	[7]	[7]	[7]
[8]	[11]	[11]	[11]	[11]	[11]	×	[8]	[8]	[8]	[8]	[8]	[7]	[7]	[7]
[9]	×	×	×	×	×	×	[9]	[9]	[9]	[9]	[9]	[9]	[9]	[9]
[10]	[11]	[11]	×	[11]	[14]	×	[8]	[8]	[8]	[8]	[10]	[9]	[9]	[9]
[11]	[11]	[11]	[11]	[11]	[11]	×	[11]	[11]	[11]	[11]	[11]	×	×	×
[12]	[8]	[8]	[8]	[8]	[10]	[9]	[11]	[11]	[11]	[11]	[12]	[13]	[13]	[13]
[13]	[8]	[8]	[8]	[8]	[10]	[9]	[11]	[11]	[11]	[11]	[13]	[13]	×	[13]
[14]	[11]	[11]	[11]	[11]	[14]	×	[11]	[11]	[11]	[11]	[14]	×	×	×

Figure 2. Multiplication table for perfect neighborhood sets in trees

From Figure 2, we can now write out a system of recurrence relations in terms of these subclasses, as follows.

$$[1] = [1]\circ[8]\cup[1]\circ[10]\cup[1]\circ[11]\cup[1]\circ[12]\cup[1]\circ[13]\cup[1]\circ[14]\cup[2]\circ[10] \cup[2]\circ[14]\cup[3]\circ[12]\cup[3]\circ[13]$$

$$[2] = [2]\circ[8]\cup[2]\circ[11]\cup[2]\circ[12]\cup[2]\circ[13]\cup[4]\circ[12]\cup[4]\circ[13]$$

$$[3] = [3]\circ[8]\cup[3]\circ[11]\cup[3]\circ[14]\cup[4]\circ[10]\cup[4]\circ[14]$$

$$[4] = [4]\circ[8]\cup[4]\circ[11]$$

$$[5] = [2]\circ[2]\cup[2]\circ[5]\cup[5]\circ[2]\cup[5]\circ[5]\cup[5]\circ[8]\cup[5]\circ[11]\cup[5] \circ[12]\cup[5]\circ[13]\cup[6]\circ[12]$$

$$[6] = [4]\circ[2]\cup[4]\circ[5]\cup[6]\circ[2]\cup[6]\circ[5]\cup[6]\circ[8]\cup[6]\circ[11]\cup[6]\circ[13]$$

$$[7] = [7]\circ[7]\cup[7]\circ[8]\cup[7]\circ[9]\cup[7]\circ[10]\cup[7]\circ[11]\cup[7]\circ[12]\cup[7] \circ[13]\cup[7]\circ[14]\cup[8]\circ[12]\cup[8]\circ[13]\cup[8]\circ[14]$$

$$[8] = [8]\circ[7]\cup[8]\circ[8]\cup[8]\circ[9]\cup[8]\circ[10]\cup[8]\circ[11]\cup[10]\circ[7]\cup[10] \circ[8]\cup[10]\circ[9]\cup[10]\circ[10]\cup[12]\circ[1]\cup[12]\circ[2]\cup[12]\circ[3]\cup[12] \circ[4]\cup[13]\circ[1]\cup[13]\circ[2]\cup[13]\circ[3]\cup[13]\circ[4]$$

$$[9] = [9]\circ[7]\cup[9]\circ[8]\cup[9]\circ[9]\cup[9]\circ[10]\cup[9]\circ[11]\cup[9]\circ[12]\cup[9] \circ[13]\cup[9]\circ[14]\cup[10]\circ[12]\cup[10]\circ[13]\cup[10]\circ[14]\cup[12]\circ[6]$$

$\cup[13] \circ [6]$

$[10] = [10] \circ [11] \cup [12] \circ [5] \cup [13] \circ [5]$

$[11] = [8] \circ [1] \cup [8] \circ [2] \cup [8] \circ [3] \cup [8] \circ [4] \cup [8] \circ [5] \cup [10] \circ [1] \cup [10] \circ [2]$
$\cup[10] \circ [4] \cup [11] \circ [1] \cup [11] \circ [2] \cup [11] \circ [3] \cup [11] \circ [4] \cup [11] \circ [5]$
$\cup[11] \circ [7] \cup [11] \circ [8] \cup [11] \circ [9] \cup [11] \circ [10] \cup [11] \circ [11] \cup [12] \circ [7]$
$\cup[12] \circ [8] \cup [12] \circ [9] \cup [12] \circ [10] \cup [13] \circ [7] \cup [13] \circ [8] \cup [13] \circ [9] \cup [13]$
$\circ[10] \cup [14] \circ [1] \cup [14] \circ [2] \cup [14] \circ [3] \cup [14] \circ [4] \cup [14] \circ [7] \cup [14]$
$\circ[8] \cup [14] \circ [9] \cup [14] \circ [10]$

$[12] = [12] \circ [11]$

$[13] = [12] \circ [12] \cup [12] \circ [13] \cup [12] \circ [14] \cup [13] \circ [11] \cup [13] \circ [12] \cup [13] \circ [14]$

$[14] = [10] \circ [5] \cup [14] \circ [5] \cup [14] \circ [11]$

To illustrate this, a tree-subset pair of class [4] can be read as follows: a tree-subset pair (T, S) which is of class [4] can be obtained only by composing a pair (T_1, S_1) of class [4] with a pair (T_2, S_2) of class [8] or by composing a pair (T_1, S_1) of class [4] with a pair (T_2, S_2) of class [11].

To prove the correctness of this dynamic programming algorithm for computing $\theta(T)$ for any tree T, we would have to prove a theorem asserting that each of these recurrences are correct. Space limitations prevent us from doing this here, but it is easy to do. It is even easier to verify the correctness of Figure 2, which can be done by inspection. The final step in specifying a θ-algorithm is to define the initial vector. In this case, for trees, the only basis graph is the tree with single vertex K_1. We need to know the minimum cardinality of a set S in a class of type [1] to [14] in the graph K_1, if any exists. It is easy to see that the initial vector is $[-,-,-,1,-,-,-,-,-,-,-,0,-,-]$ where '$-$' means undefined.

We now have all the ingredients for a θ-algorithm, where the input is the parent array $parent[1\ldots p]$ for the input tree of order p and where the output is the 14-tuple corresponding to the root (i.e., vertex 1) of T which is computed repeatedly by applying the recurrence system to each vertex in the parent array, with the initial vector being associated with every vertex in the parent array as the computation begins. The basic structure for the algorithm is a simple iteration.

```
procedure θ;

for i:=1 to p do
    initialise vector [1 ... 14] to
    [-, - , -, 1, -, -, -, -, -, -, -, 0, -, -];

for j:=p downto 2 do
begin
  k:=parent[j];
  combine(vector,k,j);
end;

θ(T) := min {vector[1,1], vector[1,2], vector[1,3], vector[1,4],
            vector[1,5], vector[1,7], vector[1,8], vector[1,9],
            vector[1,10], vector[1,11],};
end; {θ}
```

The combine procedure is derived directly from the recurrence system: For example, the fourth and tenth components of the 14-tuple corresponding to vector k are given, respectively, by

$$vector[k,4] = \min\{vector[k,4]+vector[j,8], vector[k,4]+vector[j,11]\};$$

$$vector[k,10] := min\{vector[k,10] + vector[j,11],$$

$$vector[k,12] + vector[j,5], vector[k,13] + vector[j,5]\};$$

It is clear that procedure θ has linear execution time.

References

[1] B. Bollobás and E.J. Cockayne, Graph-theoretic parameters concerning domination, independence and irredundance, *J. Graph Theory*, **3** (1979), 241–249.

[2] G.H. Fricke, T.W. Haynes, S. Hedetniemi, S.T. Hedetniemi, and M.A. Henning, Perfect Neighborhoods in Graphs, submitted for publication.

[3] T.W. Haynes, S.T. Hedetniemi, and P.J. Slater, *Domination in Graphs: the Theory*, Marcel-Dekker, Inc., 1996.

[4] O. Ore, *Theory of graphs. Amer. Math. Soc. Transl.* **38** (Amer. Math. Soc., Providence, RI, 1962), 206–212.

[5] T.V. Wimer, Linear algorithms on k-terminal graphs, Ph.D thesis, Clemson University, 1987.