ELSEVIER

Discrete Mathematics 138 (1995) 229–246

DISCRETE MATHEMATICS

Nearly perfect sets in graphs

Jean E. Dunbar[a,*], Frederick C. Harris, Jr[b], Sandra M. Hedetniemi[b], Stephen T. Hedetniemi[b], Alice A. McRae[b], Renu C. Laskar[c]

[a] *Department of Mathematics, Converse College, Spartanburg, SC 29302, USA*
[b] *Department of Computer Science, Clemson University, Clemson, SC 29634, USA*
[c] *Department of Mathematics, Clemson University, Clemson, SC 29634, USA*

Received 7 July 1993; revised 26 April 1994

Abstract

In a graph $G=(V,E)$, a set of vertices S is nearly perfect if every vertex in $V-S$ is adjacent to at most one vertex in S. Nearly perfect sets are closely related to 2-packings of graphs, strongly stable sets, dominating sets and efficient dominating sets. We say a nearly perfect set S is 1-minimal if for every vertex u in S, the set $S-\{u\}$ is not nearly perfect. Similarly, a nearly perfect set S is 1-maximal if for every vertex u in $V-S$, $S\cup\{u\}$ is not a nearly perfect set. Lastly, we define $n_p(G)$ to be the minimum cardinality of a 1-maximal nearly perfect set, and $N_p(G)$ to be the maximum cardinality of a 1-minimal nearly perfect set. In this paper we calculate these parameters for some classes of graphs. We show that the decision problem for $n_p(G)$ is NP-complete; we give a linear algorithm for determining $n_p(T)$ for any tree T; and we show that $N_p(G)$ can be calculated for any graph G in polynomial time.

1. Introduction

Let $G=(V,E)$ be a graph. We say that a set S of vertices in V is a *P-set* if S has property P, for some property under consideration. Frequently we say that a set S has property P if the subgraph of G induced by S, denoted $\langle S\rangle$, has some property P, e.g. $\langle S\rangle$ is acyclic, or planar or has degree at most k for some non-negative integer k. Other properties, however, are not defined in terms of induced subgraph structures. For example, S is a *dominating set* if every vertex in $V-S$ is adjacent to at least one vertex in S or S in an *irredundant set* if for every vertex u in S, $N[u]-N[S-\{u\}]$ is not empty. Most optimization problems in graph theory are concerned with finding either the minimum or maximum cardinality of a P-set in a graph G. However, recent research has studied the spectrum of cardinalities of a wide variety of minimal and

* Corresponding author.

SSDI 0012-365X(94)00205-3

maximal P-sets in a graph. But care should be taken in defining a minimal or a maximal P-set. There are usually two definitions given for each term.

(1) A P-set S is maximal if no proper superset S' of S is a P-set.

(1′) A P-set S is maximal if for every vertex u in $V-S$, $S\cup\{u\}$ is not a P-set.

(2) A P-set S is minimal if no proper subset S' of S is a P-set.

(2′) A P-set S is minimal if for every vertex u in S, the set $S-\{u\}$ is not a P-set.

A property P of a graph is *hereditary* if whenever a set S has property P, every proper subset S' of S also has property P. For example, the properties of being an independent set of vertices (no two vertices in the set are adjacent) or an irredundant set of vertices are hereditary. It is easy to see that definitions (1) and (1′) of maximal P-sets are equivalent if P is a hereditary property.

A property P is called *super-hereditary* (or *expanding*) if whenever a set S has property P, every superset S' of S also has property P. For example the properties of being a dominating set or a vertex cover are super-hereditary. It is also easy to see that the definitions (2) and (2′) of minimal P-sets are equivalent for super-hereditary properties P.

Bollobás et al. [1] defined new notions of minimality and maximality as follows. A set of vertices S in V is a *k-minimal P-set* if the removal of any set X of $k\leqslant|S|$ vertices from S followed by the addition of any $l<k$ vertices results in a set that does not have property P.

Similarly a set of vertices S in V is a *k-maximal P-set* if the addition of any set X of k vertices, followed by the removal of any $l<k$ vertices does not result in a P-set. It is easy to see that 1-minimal P-sets are minimal P-sets according to definition (1′) and 1-maximal P-sets are maximal P-sets according to definition (2′). It is also easy to see that there are 1-minimal P-sets which are not 2-minimal P-sets, e.g. in Fig. 1 each set of solid vertices is 1-minimal but the set in Fig. 1(a) is not a 2-minimal set, since if we remove the two endvertices and then add the center vertex to S, we obtain another dominating set.

The study of k-minimal and k-maximal sets is still very new; not much is known about these sets. The reader is referred to [1, 4, 5]. The following property provides an example for which the definition of minimal or maximal must be given carefully.

2. Definition of nearly perfect sets

Let $G=(V,E)$ be a graph and let u be a vertex in V. The *open neighborhood of u* is defined as the set of vertices v adjacent to u; i.e., $N(u)=\{v\,|\,uv\in E\}$. The *closed*

(a) (b)

Fig. 1.

Table 1

	$\forall u \in V-S$	$\forall u \in V$
$\lvert N(u)\cap S\rvert \leqslant 1$ Maximal sets	Nearly perfect n_p	Total nearly perfect $n_{tp} \leqslant N_{tp}$
$\lvert N[u]\cap S\rvert \leqslant 1$ Maximal sets	Nearly perfect n_p	Independent nearly perfect (strong stable set or 2-packing) $p_2 \leqslant P_2$
$\lvert N(u)\cap S\rvert = 1$	Perfect domination γ_p	Total perfect domination $\gamma_{tp} \leqslant \Gamma_{tp}$
$\lvert N[u]\cap S\rvert = 1$	Perfect domination γ_p	Efficient domination $\gamma_e \leqslant \Gamma_e$
$\lvert N(u)\cap S\rvert \geqslant 1$ Minimal sets	Domination $\gamma \leqslant \Gamma$	Total domination $\gamma_t \leqslant \Gamma_t$
$\lvert N[u]\cap S\rvert \geqslant 1$ Minimal sets	Domination $\gamma \leqslant \Gamma$	Domination $\gamma \leqslant \Gamma$

neighborhood of u is $N[u]=N(u)\cup\{u\}$. In a graph G, a set of vertices S is *nearly perfect* if for every vertex $u\in V-S$, $|N(u)\cap S|\leqslant 1$; i.e., every vertex $u\in V-S$ is adjacent to at most one vertex in S.

Nearly perfect sets are closely related to 2-packings of graphs, strongly stable sets, dominating sets and efficient dominating sets, as indicated in Table 1.

Table 1 first appeared in a paper by Cockayne et al. [2], which focused on perfect dominating sets, and in particular on families of graphs whose only perfect dominating set is the entire vertex set V. Nearly perfect sets, although first defined in [2], were not studied there. In this paper we initiate the study of nearly perfect sets in graphs.

Notice that for any graph G, the empty set ϕ of vertices, any set $S=\{u\}$ consisting of a single vertex, and the set V are all nearly perfect sets. Thus, using definition (1), the only minimal nearly perfect set in any graph is the empty set. And using definition (2), the only maximal nearly perfect set in any graph is the set V.

However, using definitions (1′) and (2′), a graph can have nontrivial minimal and maximal nearly perfect sets. For example, in the graph C_4 in Fig. 2 the set $\{1,2\}$ is a 1-maximal nearly perfect set; and in Fig. 3 the set $\{1,2,5\}$ is a 1-minimal nearly perfect set in the graph G.

Define $n_p(G)$ to equal the minimum cardinality of a 1-maximal nearly perfect set, and $N_p(G)$ to equal the maximum cardinality of a 1-minimal nearly perfect set in G. Notice that $n_p(G)$ is well-defined since V is always a 1-maximal nearly perfect set; i.e., $n_p(G)\leqslant|V|$. In particular, $n_p(K_n)=1$, for $n\geqslant 3$ and $n_p(\bar{K}_n)=n$.

Also $N_p(G)$ is well-defined since the empty set is always a 1-minimal nearly perfect set; i.e., $N_p(G)\geqslant 0$. In particular, $N_p(K_n)=n$ for $n\geqslant 3$, and $N_p(\bar{K}_n)=0$.

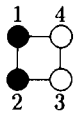

Fig. 2.

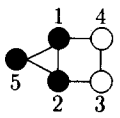

Fig. 3

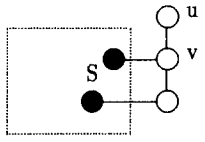

Fig. 4.

3. Characteristics of the parameter $n_p(G)$

The first result characterizes the property of 1-maximality in a nearly perfect set.

Lemma 1. *A nearly perfect set S in a graph $G=(V,E)$ is 1-maximal if and only if every vertex u in $V-S$ is adjacent to a vertex $v\neq u$ in $V-S$ which is adjacent to exactly one vertex in S.*

Proof. The characterizing property given above is illustrated in Fig. 4. Suppose a set S is 1-maximal nearly perfect. Let u be an element of $V-S$ and assume that every vertex w adjacent to u in $V-S$ is not adjacent to any vertex in S. Then every element outside of the set $S\cup\{u\}$ is adjacent to at most one element in the set $S\cup\{u\}$. Thus, the set $S\cup\{u\}$ is nearly perfect. This contradicts the assumption that S is 1-maximal nearly perfect.

On the other hand, if S is nearly perfect and satisfies the property above, then for any vertex u in $V-S$, there is a vertex u' in $(V-S)\cup N(u)$, with u' adjacent to some vertex v in S. Thus, $S\cup\{u\}$ is not a nearly perfect set, since u' is adjacent to the two vertices u and v in $S\cup\{u\}$. Since u was chosen arbitrarily, we know that the union of S with any vertex in $V-S$ will not be a nearly perfect set. Thus, S is a 1-maximal nearly perfect set. □

Using Lemma 1, the values of $n_p(G)$ for some classes of graphs are easily determined. For example, the complete bipartite graph $K_{m,n}$ (for $m,n\geqslant 2$) has a smallest 1-maximal nearly perfect set of size two. One such set can be found by selecting one vertex from each of the partite sets of the vertex set. Since such a set is nearly perfect, it follows from Lemma 1 that the set is also 1-maximal. The following lemma will be used to calculate the parameter $n_p(G)$ for paths and cycles.

Lemma 2. *Suppose $G=(V,E)$ is a path or a cycle with order at least four and that S is a subset of the vertices of G. Then S is a 1-maximal nearly perfect set if and only if*

(i) *the subgraph spanned by $V-S$ is a disjoint union of K_2 graphs, and*

(ii) *every vertex in $V-S$ is adjacent to exactly one vertex of S.*

Proof. Suppose S is a 1-maximal nearly perfect set and let v be an element of $V-S$. By Lemma 1, we know there is an element $w\neq v$ in $V-S$ with w adjacent to v and w adjacent to one vertex x of S. Since the maximum degree of G is two, $N(w)=\{v,x\}$. Further since w is in $V-S$, Lemma 1 guarantees that a neighbor of w must lie outside of S and be adjacent to an element of S. Since all neighbors of w are accounted for, this element must be v. So v must be adjacent to a vertex, say y, in S. Note that $y\neq x$. Since the maximum degree in G is two, we must have that $N(v)=\{w,y\}$. Since v was an arbitrary element of $V-S$, we must have that $\langle V-S\rangle$ consists of disjoint copies of K_2 and that every element of $V-S$ is adjacent to exactly one element in S.

Conversely, if S is a subset of V for which properties (i) and (ii) hold, then property (ii) guarantees that S is a nearly perfect set. Moreover, any vertex of $V-S$ will be adjacent to another vertex of $V-S$ which is adjacent to a vertex in S. Thus by Lemma 1 we know that S is a 1-maximal nearly perfect set. □

Proposition 3. *If P_n is a path with n vertices, then*

$$n_p(P_n)=\begin{cases} k+2 & \text{if } n=3k,\\ k+1 & \text{if } n=3k+1,\\ k+2 & \text{if } n=3k+2.\end{cases}$$

Proof. Let P be the path $p_1p_2\ldots p_n$. The proof is by induction on n. Clearly the result holds for paths with $n=1$, 2 *or* 3 vertices. Suppose $n>3$ and assume the result holds for all paths with less than n vertices.

Case 1: $n=3k$. Let $S=\{p_1,p_4,p_7,\ldots,p_{n-2},p_{n-1},p_n\}$. Note first that $n-2=3(k-1)+1$. Since $V-S$ is a disjoint collection of K_2 graphs and every vertex of $V-S$ is adjacent to exactly one vertex of S, we know by Lemma 2 that S is a 1-maximal nearly perfect set. Further note that $|S|=(k-1)+3=k+2$. Suppose there exists a 1-maximal nearly perfect set T with cardinality less than the cardinality of S. Let $j=\max\{i\mid p_i\in V-T\}$. Since every element of $V-T$ must have degree two, we know $j<n$. Further, $j>3$, for if not then with $j\leqslant 3$ we would know that all vertices to the right of p_j were in T and at least one vertex to the left of p_j must be in T. Thus under this assumption we would have $|T|\geqslant 3k-2\geqslant k+2=|S|$. The last inequality holds because $k\geqslant 2$. So we must have $3<j<n$. Since T is 1-maximal nearly perfect, we must have p_{j-1} in $V-T$, and p_{j-2} in T. Let $U=\{p_{j-2},p_{j-1},p_j\}$, and consider the graph G' obtained by adding an edge $p_{j-3}p_{j+1}$ to the graph $G-U$. See Fig. 5. Since p_{j-2} is in T and p_{j+1} is in T, the new edge allows the same adjacencies for p_{j-3} in G' as in G. Then $T-U$ is a nearly perfect set in G' (which is P_{n-3}) and the cardinality of $T-U$ is $|T|-1<(k+2)-1=(k-1)+2$, contradicting the inductive hypothesis on P_{n-3}.

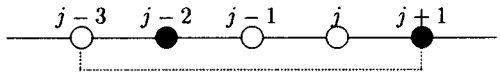

Fig. 5.

Case 2: $n=3k+1$. Let $S=\{p_1,p_4,p_7,\ldots,p_n\}$. Note that $|S|=k+1$, and, as before, S is a 1-maximal nearly perfect set. In exactly the same way as in Case 1, we may assume T has cardinality less than that of S and is a 1-maximal nearly perfect set in the path. It is immediate to see that a subset of the path P_{n-3} can be found which is 1-maximal nearly perfect and which contradicts the inductive hypothesis on P_{n-3}.

Case 3: $n=3k+2$. Let $S=\{p_1,p_4,p_7,\ldots,p_{3k+1},p_{3k+2}\}$. Note that $|S|=k+2$, and that S is a 1-maximal nearly perfect subset of the path. The proof that S is of smallest size uses the same argument as that for the first two cases. □

Proposition 4. *If C_n is a cycle with n vertices, then*

$$n_p(C_n)=\begin{cases} k & \text{if } n=3k,\\ k+1 & \text{if } n=3k+1,\\ k+2 & \text{if } n=3k+2.\end{cases}$$

The proof is similar to that of Proposition 3 and is omitted.

For some graphs the smallest 1-maximal nearly perfect set must include all vertices in the graph. One such example already mentioned is $\bar{K}_n$. Another easily verified example is $K_{1,n}$ (a star). In fact as the next two results show, stars are the only connected graphs for which $n_p(G)=n$.

Theorem 5. *For any tree T with n vertices, $n_p(T)=n$ if and only if T has diameter at most 2; i.e., T is a star.*

Proof. Let T be a tree with n vertices and diameter at most 2 and let S be an n_p-set, that is, S is a minimum cardinality 1-maximal nearly perfect set. Assume $|S|<n$ and let x be a vertex in $V-S$. Then there is a vertex $y\in(V-S)\cap N(x)$ such that y is adjacent to a vertex y' in S. Since $y\notin S$, y must be adjacent to some vertex z in $V-S$ and z must be adjacent to a vertex z' in S. Now y' is not adjacent to z and z' is not adjacent to y, for if so a cycle would be formed. Thus z', z, y, y' constitutes a shortest path of length 3 from z' to y', contradicting the diameter of T.

Conversely let T be a tree with diameter at least 3. For this class of trees we will prove by induction on n that $n_p(T)<n$. The only tree with diameter at least 3 and $n=4$ is P_4 and by Proposition 3, $n_p(P_4)=2$.

So let T be a tree with $n>4$ vertices and diameter d at least 3. Let u be an endvertex such that $T-\{u\}$ has diameter at least 3. Such a u must exist as follows. Let $w-v$ be any path of length d (w and v are necessarily endvertices). If $d>3$, then $diameter(T-\{u\})\geqslant 3$. If $d=3$ (and $n>4$), then there is at least one endvertex u, not on this $w-v$ path. Therefore, $T-\{u\}$ has diameter at least 3.

Let $T-\{u\}$ have diameter at least 3 and $T'=T-\{u\}$. Then by the inductive hypothesis $n_p(T')<n-1$. Let S be an n_p-set of T'.

Case 1: *u is adjacent to a vertex in S.* In this case, by Lemma 1, $S\cup\{u\}$ is a 1-maximal nearly perfect set of T and therefore $n_p(T)<n$.

Case 2: *u is adjacent to a vertex in $V(T')-S$.* Let u be adjacent to v.

Case 2a: *v is adjacent to a vertex in S.* In this case S is an n_p-set in T. Hence $n_p(T)<n-1<n$.

Case 2b: *v is not adjacent to any vertex in S.* By Lemma 1, v must be adjacent to a vertex w in $V(T')-S$, which in turn is adjacent to exactly one vertex in S. In this case $S\cup\{u\}$ is a nearly perfect set in T and, by Lemma 1, is a 1-maximal nearly perfect set. Hence $n_p(T)<n$. □

Theorem 6. *For any connected graph G with n vertices, $n_p(G)=n$ if and only if G is a star.*

Proof. If G is a star with n vertices then $n_p(G)=n$ by Theorem 5. It remains to show that if $n_p(G)=n$ then G is a star.

Case 1: *G is acyclic.* Since G is connected, G is a tree, and again by Theorem 5, G is a star.

Case 2: *G contains a triangle.* Let the vertices of $G=(V,E)$ be ordered such that v_1, v_2, and v_3 form a triangle. In this case we can construct a 1-minimal nearly perfect set S in G for which $|S|<n$, which implies that $n_p(G)<n$. We do this as follows. Initially let $S=\{v_1\}$. Clearly, S is a nearly perfect set. Furthermore, neither $S\cup\{v_2\}$ nor $S\cup\{v_3\}$ is a nearly perfect set. We construct a 1-maximal nearly perfect set which contains neither v_2 nor v_3 by executing the following:

```
Procedure Make 1-maximal
  Input: A nearly perfect set S
  Output: A 1-maximal nearly perfect set S
Begin
  while ∃w∈V−S−{v2, v3} for which S∪{w} is nearly perfect do
    S←S∪{w}
  od
End Make 1-maximal.
```

Clearly when this procedure has finished, the resulting set S is nearly perfect, contains neither v_2 nor v_3, and for every vertex u in $V-S$, $S\cup\{u\}$ is not a nearly perfect set. Thus, S is a 1-maximal nearly perfect set, and $|S|<n$; i.e., $n_p(G)<n$.

Case 3: *G contains no triangles but contains a cycle of length four.* Let $C=\{v_1,v_2,v_3,v_4\}$ be the vertices in cyclic order of such a cycle. As in the previous case, we show that we can construct a 1-maximal nearly perfect set S with $|S|<n$, which implies that $n_p(G)<n$. We do this as follows. Initially let $S=\{v_1,v_4\}$. Since G has no triangles, S is a nearly perfect set. We can now execute Procedure Make

1-maximal (as defined in Case 2) and construct a 1-maximal nearly perfect set S with $|S|<n$ (the final set will contain neither v_2 nor v_3).

Case 4: *G contains no triangles and no cycles of length four.* Let $C=\{v_1,v_2,v_3,v_4,\ldots,v_m\}$ be the vertices, in cyclic order, of a cycle in G, where $m\geqslant 5$. Once again we construct a 1-maximal nearly perfect set S with $|S|<n$. Initially let $S=\{v_1,v_4\}$. If S is a nearly perfect set, then we can again apply Procedure Make 1-maximal and construct a 1-maximal nearly perfect set S with $|S|<n$.

If S is not a nearly perfect set then there must be a vertex, say w, adjacent to both v_1 and v_4. Note that there can only be one such vertex, w, else G contains a cycle of length four. Let $S=\{v_1,v_4,w\}$. It follows that S must be a nearly perfect set since no vertex can be adjacent to two of these three vertices (note that if a vertex is adjacent to v_1 and w, or v_4 and w then G must have a triangle). Thus we can apply Procedure Make 1-maximal to S and construct a 1-maximal nearly perfect set S with $|S|<n$. □

Proposition 7. *For any graph* $G=(V,E)$ $n_p(G)=1$ *if and only if there is a vertex* u *in* V *such that for every vertex* $v\neq u$ *there is a path of length two from* u *to* v.

Proof. Clearly if such a vertex u exists then the set $S=\{u\}$ is a 1-maximal nearly perfect set, and hence $n_p(G)=1$.

Conversely, let $n_p(G)=1$ and let $S=\{u\}$ be a 1-maximal nearly perfect set. By definition, for every vertex $v\neq u$ the set $S'=\{u\}\cup\{v\}$ is not a nearly perfect set. This means that there is a third vertex w which is adjacent to both u and v; i.e., that there is a path of length two from u to v. □

Corollary 8. *For complete graphs* K_n *and wheels,* $W_n\cong K_1+C_n$, $n\geqslant 3$, $n_p(K_n)=1$ *and* $n_p(W_n)=1$.

4. Complexity issues for n_p

In this section several complexity results are given. A linear time algorithm is given which computes the value $n_p(T)$ for any tree T. In the general case, however, the decision problem for $n_p(G)$ is NP-complete, even when restricted to bipartite or chordal graphs.

For the first result, a dynamic programming style algorithm is constructed using a blend of methodologies developed by Wimer in 1988 [10], as illustrated in [11, 7], and by Mitchell in 1977 [9], as illustrated in [3].

The class of (rooted) trees can be constructed recursively from copies of a single vertex, K_1, using only one rule of composition. This rule combines two rooted 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 tree. This is denoted by $(T,r_1)=(T_1,r_1)\circ(T_2,r_2)$. In particular, if S is a 1-maximal nearly perfect set in T, then S splits into two subsets S_1 and S_2 according

Table 2

	[1]	[2]	[3]	[4]	[5]	[6]	[7]
Property							
(1) $r \in S$	√	×	×	×	×	×	×
(2) S is 1-maximal nearly perfect in T	√	√	√	×	×	×	×
(3) All vertices in $T-r$ have property $P(S)$	√	√	√	√	×	√	×
(4) $\lvert N[r] \cap S \rvert = 0$	×	×	√	×	√	√	√
(5) r has property $P(S)$	√	√	√	×	√	×	×

to this composition/decomposition. In order to construct an algorithm for $n_p(T)$ it is necessary to characterize the classes, TS, of possible tree–subset pairs (T, S) which can occur in the process of decomposing a tree T and a 1-maximal nearly perfect set S.

For a given set of vertices S, we say that a vertex v has *Property* $P(S)$ if either $v \in S$ or there is a vertex u in the intersection of $N(v)$ and $V-S$ such that $\lvert N(u) \cap S \rvert = 1$.

Next we define the collection of possible tree–subset pairs TS as the set of all ordered pairs (T, S) which satisfy these properties:

- T is a rooted tree with root r,
- S is a nearly perfect set in T, and
- if r is not in S, then all vertices of T, except possibly those in $N[r]$, have property $P(S)$, otherwise all vertices of T have property $P(S)$.

Consider next the seven subclasses of TS defined by Table 2. Each subclass is defined by the presence (indicated with √) or absence (indicated with ×) of the five properties listed in the table. The symbol r denotes the root of T. We will denote these subclasses with $[i]$, $i = 1 \ldots 7$. Thus for example, subclass [2] is the set of all tree–set pairs (T, S) in which the root is not a member of S, S is a 1-maximal nearly perfect set in T, all vertices in $T-r$ have property $P(S)$, r is adjacent to some element of S, and r has property $P(S)$. Finally, note that the subclasses [4–7] are exactly those in which S is not a 1-maximal nearly perfect set in T.

Lemma 9. *TS is the disjoint union of the seven classes defined in Table 2.*

Proof. Since each class is defined by a unique set of boolean values, they must be disjoint. It remains to show that every member of TS is a member of some subclass. There are five properties in Table 2. However property 2: 'S is 1-maximal nearly perfect in T' is merely the intersection of property 3 and property 5. Thus this property is determined by the other two. There are exactly 16 boolean assignments to the remaining four properties. Note, however, if property 1 holds (r is an element of S), then properties 3 and 5 must hold while property 4 cannot hold. Thus the seven remaining combinations which have property 1 cannot occur. Further we note that if property 1 does not hold and property 4 does not hold, then property 3 must hold. So

Table 3

	[1]	[2]	[3]	[4]	[5]	[6]	[7]
[1]	[1]	×	[1]	×	[1]	×	×
[2]	×	[2]	[2]	[2]	×	[2]	×
[3]	[2]	[3]	[3]	[5]	×	[5]	×
[4]	×	[2]	[4]	[2]	×	[4]	×
[5]	[2]	[5]	[5]	[5]	×	[5]	×
[6]	[4]	[3]	[6]	[5]	×	[7]	×
[7]	[4]	[5]	[7]	[5]	×	[7]	×

two more combinations are eliminated. Hence, there can be at most seven nonempty subclasses of TS. □

These subclasses and the following composition operation will be used in the design of the algorithm. Let $T_1=(V_1,E_1)$ and $T_2=(V_2,E_2)$ be trees with roots r_1 and r_2, respectively, and let $V=V_1\cup V_2$ and $E=E_1\cup E_2\cup(r_1,r_2)$. Then the composition of T_1 and T_2, denoted $T_1\circ T_2$, is the tree $T=(V,E)$ with root r_1. For two tree–set pairs (T_1,S_1) and (T_2,S_2), we will denote the pair $(T_1\circ T_2,S_1\cup S_2)$ with $(T_1,S_1)\circ(T_2,S_2)$.

Given members $(T_1,S_1)\in[n]$ and $(T_2,n_2)\in[m]$ we can determine the class, if any, to which the pair $(T_1,S_1)\circ(T_2,S_2)$ must belong. Table 3 summarizes these compositions. The symbol × indicates that the resulting tree–set pair, though defined, is not a member of TS.

We can now derive a set of recurrence equations in terms of these subclasses. By Table 3 we know that

$$[1]=[1]\circ[1]\cup[1]\circ[3]\cup[1]\circ[5]$$
$$[2]=[2]\circ[2]\cup[2]\circ[3]\cup[2]\circ[4]\cup[2]\circ[6]\cup[3]\circ[1]\cup[4]\circ[2]\cup[4]\circ[4]\cup[5]\circ[1]$$
$$[3]=[3]\circ[2]\cup[3]\circ[3]\cup[6]\circ[2]$$
$$[4]=[4]\circ[3]\cup[4]\circ[6]\cup[6]\circ[1]\cup[7]\circ[1]$$
$$[5]=[3]\circ[4]\cup[3]\circ[6]\cup[5]\circ[2]\cup[5]\circ[3]\cup[5]\circ[4]\cup[5]\circ[6]\cup[6]\circ[4]\cup[7]\circ[2]\cup[7]\circ[4]$$
$$[6]=[6]\circ[3]$$
$$[7]=[6]\circ[6]\cup[7]\circ[3]\cup[7]\circ[6]$$

For $1\leqslant i\leqslant 7$, let $a_i(T)$ be defined as follows: $a_i(T)=\min\{|S|\,|\,S\subseteq V,(T,S)\subseteq[i]\}$.

Then, for example, $a_1(T)=\min\{a_1(T_1)+a_1(T_2),a_1(T_1)+a_3(T_2),a_1(T_1)+a_5(T_2)\}$

To prove the correctness of this dynamic programming algorithm for computing $n_p(T)$ for any tree T, we would have to prove a theorem asserting that each of these

recurrences is correct. These details are straightforward and are omitted here. The final step in specifying an algorithm for an n_p-set is to define the 'initial vector'. In this case, for trees, the only basis graph is the tree with a single vertex, K_1. We need to know the minimum cardinality of a set S in a class of type [1]...[7] in the graph K_1, if any exists. It is easy to see that the initial vector is $(1, \infty, \infty, \infty, \infty, 0, \infty)$ which means that for the graph K_1:

one tree–set pair of class [1] exists and its cardinality is 1;
one tree–set pair of class [6] exists and its cardinality is 0;
no tree–set pairs of class [2–5] or class [7] exist.

We now have all of the ingredients for an n_p-set algorithm, where the input is the parent array Parent$[1 \ldots N]$ for the input tree T, and where the output is $n_p(T)$, which is computed by repeatedly applying the recurrence system to each vertex in the parent array, with the initial vector $(1, \infty, \infty, \infty, \infty, 0, \infty)$ bing associated with every vertex in the parent array as the computation begins.

The basic structure for the algorithm is a simple iteration:

```
Algorithm n_p tree
  Input: the parent array Parent[1...N] for an arbitrary tree T
  Output: n_p(T)
Begin
  for i←1 to N do
    initialize Vector [i, 1...7]←(1, ∞, ∞, ∞, ∞, 0, ∞)
  od
  for j←N to 2 do
      k←Parent[j]
      Combine (Vector, k, j)
  od
  n_p(T)←min(Vector[1,1], Vector[1,2], Vector[1,3])
End n_p−tree.
```

The Combine procedure is derived directly from the recurrence system:

```
Procedure Combine (Vector, k, j)
Begin
  Vector[k,1]←min(Vector[k,1]+Vector[j,1], Vector[k,1]+Vector[j,3],
    Vector[k,1]+Vector[j,5])
  etc...
End Combine
```

Theorem 10. *$n_p(T)$ can be computed for any tree T with N vertices in $O(N)$ time.*

Proof. The result is immediate by noting that for Algorithm n_p tree, the steps in each for-loop can be performed in constant time. It follows that if the tree has N vertices, then the algorithm operates in $O(N)$ time. □

To show that the decision problem for arbitrary graphs is NP-complete, we need to use a known NP-completeness result, called Exact Three Cover (X3C) which is defined as follows.

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'?

FACT: X3C is NP-complete [6].

n_p SET

Instance: A graph $G=(V,E)$ and a positive integer k.

Question: Is there a 1-maximal nearly perfect set S contained in V with $|S| \leqslant k$?

Theorem 11. *n_p SET is NP-complete, even when restricted to bipartite graphs.*

Proof. A nondeterministic algorithm could guess a subset $S \subseteq V$ of k vertices and, in deterministic polynomial time, verify whether or not S is 1-maximal nearly perfect. Hence the problem is in NP. In order to show NP-completeness, we reduce an equivalent version of X3C (in which every element of X lies in at least two subsets in C) to the n_p *SET* problem as follows.

Let $X=\{x_1, x_2, \ldots, x_{3q}\}$ and $C=\{C_1, C_2, \ldots, C_m\}$ be an instance of X3C, where q and m are positive integers and each C_i is a 3-element subset of X. We will show how to construct an undirected graph $G=(V,E)$ and a positive integer k such that C contains an exact cover for X if and only if $n_p(G) \leqslant k$. First construct a collection of subgraphs, G_i, $1 \leqslant i \leqslant m$, by adding a P_2 to a non-center vertex of a $K_{1,3}$ graph and label the vertices as shown in Fig. 6. Next construct a set of $3q$ paths with 2 vertices and label the vertices of each path as x_i, y_i. Add a set of edges $E=\{(x_i, c_j) \mid x_i \in C_j\}$ (cf. Fig. 7). Finally, let $k=3m$. Clearly the above reduction can be performed in time that is polynomial in m and q, and it is easy to verify that this graph G is bipartite. We now show that the set C contains an exact 3-cover for X if and only if $n_p(G) \leqslant k$.

(*Only if*) Suppose that C contains an exact 3-cover $C' \subseteq C$. Construct a set $S = \cup V_i$, $i=1 \ldots 3$, where

$$V_1 = \{a_i \mid 1 \leqslant i \leqslant m\},$$

$$V_2 = \{f(G_i) \mid 1 \leqslant i \leqslant m\},$$

$$V_3 = \{g(G_i) \mid 1 \leqslant i \leqslant m\},$$

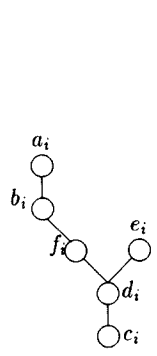

Fig. 6.

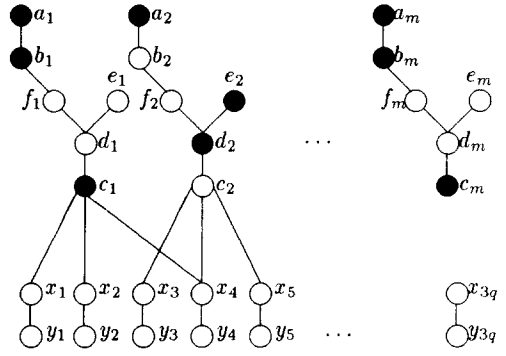

Fig. 7.

where $f(G_i)$ and $g(G_i)$ are defined as follows:

$$f(G_i)=\begin{cases} c_i & \text{if } C_i\in C', \\ d_i & \text{otherwise,} \end{cases}$$

$$g(G_i)=\begin{cases} b_i & \text{if } C_i\in C', \\ e_i & \text{otherwise.} \end{cases}$$

In Fig. 7 the solid vertices are in S. It is easy to verify that S is a 1-maximal nearly perfect set in G and since the cardinality of S is $3m$, clearly $n_p(G)\leqslant 3m$.

(If) Now suppose $n_p(G)\leqslant k$ and let S be a 1-maximal nearly perfect set of vertices with cardinality $\leqslant k$.

Observation 1. $a_i\in S$. For if not, by Lemma 1, $b_i\notin S$ and $f_i\in S$. But then b_i does not satisfy Lemma 1 and S is not a 1-maximal nearly perfect set.

Observation 2. $b_i\in S$ or $d_i\in S$. This observation is true for the same reasons as in Observation 1, for if not, we either violate Lemma 1 or the fact that S is 1-maximal nearly perfect.

Observation 3. $e_i\in S$ or $f_i\in S$ or $c_i\in S$. Otherwise e_i does not satisfy Lemma 1 and S is not a 1-maximal nearly perfect set.

Notice that each subgraph G_i must have three vertices in S, thus accounting for all $3m$ vertices in S.

Now define $C'\subseteq C$ as $C'=\{C_i \mid c_i\in S\}$. It follows from the above observations that C' is an exact 3-cover for X. If not, then some x_i is not covered by any set in C' or some $x_i\in X$ is covered by more than one set in C'.

If x_i is covered by more than one set in C', then x_i is outside S and is adjacent to two elements of S. Thus S is not a nearly perfect set. If for some x_i, x_i is not adjacent to any c_i in S, then y_i could be added to S without altering its nearly perfect status. In this case S would not be a 1-maximal nearly perfect set. □

Corollary 12. *n_p SET is NP-complete when restricted to chordal graphs.*

Proof. Forming a clique among the x_i vertices in the construction in Theorem 11, yields a graph G which is chordal. The same argument holds for this graph. □

The problem of finding a smallest 1-maximal nearly perfect set will be NP-hard for general graphs. For some NP-hard problems there are polynomial approximation algorithms that can find a solution within a constant factor f of an optimal solution. For others, there can be no polynomial approximation algorithms that are guaranteed to come within a given constant factor f of an optimal solution, unless P = NP. With a slight modification to the construction in Theorem 11, we can show the problem of finding a smallest 1-maximal nearly perfect set falls into this second category.

The proof is based upon a construction derived by Irving [8] which shows that independent domination cannot be approximated within a factor of f.

A graph constructed from an instance of X3C will be described so that the instance of X3C will have an exact cover if and only if the graph has a 1-maximal nearly perfect set of cardinality $3m$, where m is the number of subsets in the X3C instance. It will also be shown that if the graph has a 1-maximal nearly perfect set of cardinality $\leqslant f * 3m$, then the X3C instance has an exact cover.

Theorem 13. *Unless* P = NP, *there is no polynomial approximation algorithm A that can take as input a bipartite graph G and find a 1-maximal nearly perfect set for G of cardinality within a factor f of a smallest 1-maximal nearly perfect set.*

Proof. For a given constant f, consider the following graph constructed from an instance of X3C, which is similar to the graph constructed in Theorem 11. Construct a collection of subgraphs, G_i, $1 \leqslant i \leqslant m$, by adding a P_2 to a noncenter vertex of a $K_{1,3}$ graph and label these as shown in Fig. 6, where m refers to the number of subsets in C. Next construct a set of $(3mf-3m+1) \times 3q$ paths with 2 vertices each and label the vertices of each path as $x_{i,j}$ and $y_{i,j}$, $1 \leqslant i \leqslant 3q$, $1 \leqslant j \leqslant 3mf-3m+1$. Add a set of edges $E = \{(x_{i,j}, c_k) \mid x_i \in C_k\}$. Since f is a constant, this construction is polynomial in m and q.

Using the same argument as in Theorem 11, we can show that G has a 1-maximal nearly perfect set of cardinality $\leqslant 3m$ if and only if C has an exact cover. But we can also show that C has an exact cover if G has a 1-maximal nearly perfect set of cardinality $\leqslant 3mf$.

Let S be a 1-maximal nearly perfect set for G with $|S| \leqslant 3mf$. As before, let $C' = \{C_i \mid c_i \in S\}$. It follows that C' is an exact 3-cover for X. If some x_i had not been covered, then all $y_{i,j}$ vertices would have to be included in any 1-maximal nearly perfect set S. In this case, $|S| \geqslant 3mf+1$, a contradiction. If x_i is included in more than one set in C', then all $x_{i,j}$ vertices in G are adjacent to more than one element of S. Thus for S to be nearly perfect, all $x_{i,j}$ vertices must be included in S. Again this implies $|S| \geqslant 3fm+1$, which is a contradiction. Therefore, if $|S| \leqslant 3fm$, then C has an exact cover.

Suppose A is an algorithm that can find a 1-maximal nearly perfect set within a factor f of an optimal solution. Then algorithm A can be used to solve the decision problem for X3C. In polynomial time, a graph G is constructed as described above. This graph is given to algorithm A as input. Algorithm A finds a 1-maximal nearly perfect set with cardinality $\leqslant 3fm$ if and only if C has an exact cover. If algorithm A is polynomial, then the decision problem for X3C is polynomial as well. □

5. Characteristics of the parameter $N_p(G)$

The parameter $N_p(G)$, given by the size of a largest 1-minimal nearly perfect set, is surprisingly different from its counterpart $n_p(G)$. For example, the value of $N_p(G)$ can be calculated in linear time for any graph G. Before proving this, a useful characterizing lemma is necessary.

Lemma 14. *Every nonempty nearly perfect set S is 1-minimal if and only if the subgraph induced by S, $\langle S \rangle$, has minimum degree two.*

Proof. Let S be a nonempty, nearly perfect set in a graph G. Assume S is 1-minimal and let x be a vertex in S. Since S is 1-minimal, we know that $S-\{x\}$ is not a nearly perfect set. Thus x must be adjacent to at least two vertices in S. Since x was chosen arbitrarily, the minimum degree of $\langle S \rangle$ must be at least 2. The converse follows in exactly the same way. □

The following three corollaries are immediate and are given without proof.

Corollary 15. *$N_p(G)=n$ if and only if $\delta(G)\geqslant 2$.*

Corollary 16. *For any tree T, $N_p(T)=0$.*

Corollary 17. *For any graph G, $N_p(G)=0$ if and only if G is a forest.*

Algorithm N_p
  Input: an arbitrary graph G
  Output: $N_p(G)$ and a maximum cardinality 1-minimal nearly perfect set S
Begin
  $G'=G$
  While G' has a vertex u of degree at most 1 do
    $V(G')=V(G')-\{u\}$
  od
  $N_p(G)=|V(G')|$
  $S=V(G')$
End N_p

Theorem 18. *For any graph G, Algorithm N_p computes $N_p(G)$ and finds a maximum cardinality 1-minimal nearly perfect set S.*

Proof. The N_p-set S which results at the end of Algorithm N_p is nearly perfect. Clearly if S is empty, then S is nearly perfect, and in this case the graph G has no cycles, and is therefore a forest. Corollary 17 then shows that in this case $N_p(G)$ has been correctly calculated.

Let us assume without loss of generality, that G is connected. Assume next that the graph G has no cycle; i.e., G is a tree. One can easily prove by induction, that Algorithm N_p removes every vertex of G. Hence Algorithm N_p correctly asserts that $N_p(G)=0$. Assume therefore that G has a cycle C. Algorithm N_p cannot remove any vertex of C, regardless of the order in which the vertices of G are removed. For in order to remove a vertex of a cycle, its degree must become at most 1, which means that another vertex on the cycle must have been removed previously.

Assume therefore that S is not empty and further assume that S is not a nearly perfect set. Then there exists a vertex v in $V-S$ which is adjacent to at least two vertices, x and y, in S. Since v is in $V-S$, Algorithm N_p removed v from $V(G')$, implying that when it was removed its degree was at most 1. This implies that either x or y must have been removed from S before v, which contradicts the assumption that x and y are in S. Thus, S is nearly perfect. By Lemma 14 it follows that S is 1-minimal nearly perfect. It remains to show that S has maximum cardinality.

Assume that S is not of maximum cardinality and let M be a 1-minimal nearly perfect set with $|M|>|S|$. Let the vertices of $V(G)-S$ be ordered as $\{u_1, u_2, \ldots, u_k\}$ in the order in which they are removed by Algorithm N_p and let $w=u_i$ be the first vertex of $M-S$ which is removed by Algorithm N_p. It follows from Lemma 14 that w is adjacent to at least two vertices in M, say r and s. But w is removed by Algorithm N_p only if its degree becomes at most 1; i.e., either r or s must have been removed from $V(G')$ before w. But this contradicts the assumption that w is the first vertex of $M-S$ which is removed by Algorithm N_p. Thus S is a maximum cardinality 1-minimal nearly perfect set. □

Theorem 19. *Every graph G has a unique maximum cardinality 1-minimal nearly perfect set.*

Proof. We can assume without loss of generality that $N_p(G)\neq 0$. Let S be produced by Algorithm N_p. By Theorem 18, we know that S is a maximum cardinality 1-minimal nearly perfect set. Let N be any other maximum cardinality 1-minimal nearly perfect set of G and let w be the first vertex of $N-S$ which is removed by Algorithm N_p. Since w is in N, it must have degree at least 2 in $\langle N\rangle$, by Lemma 14. Let vertices x and y be vertices adjacent to w in $\langle N\rangle$. Consider the point in Algorithm N_p when w is removed. Vertex w can only be removed when its degree has become at most 1; i.e., either x or y has been previously removed. But this contradicts the assumption that w is the first vertex of $N-S$ that is removed by Algorithm N_p. Thus $N-S$ is empty and S is unique. □

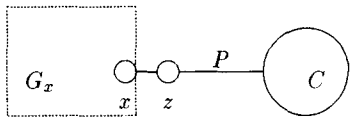

Fig. 8.

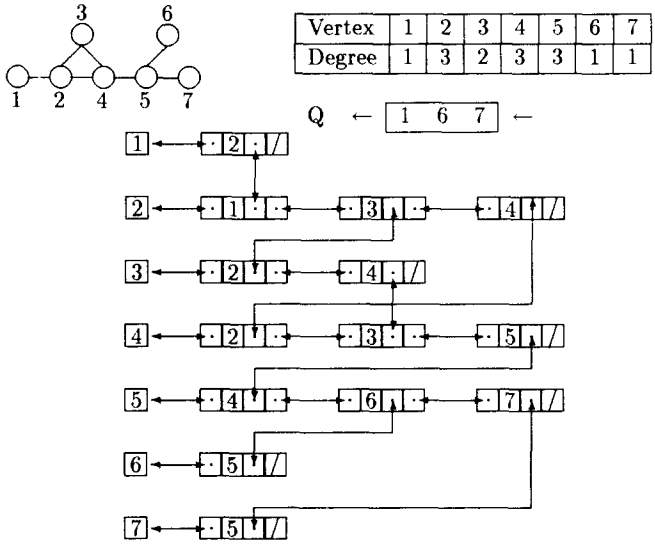

Fig. 9.

Theorem 20. *For any graph G, its unique maximum cardinality 1-minimal nearly perfect set consists of the set of all vertices which are contained in either a cycle or a path connecting two disjoint cycles.*

Proof. Let us assume without loss of generality that G is connected and that $N_p(G) \neq 0$. Using an argument similar to that used in proving Theorem 18, one can easily show that Algorithm N_p cannot remove any vertex in a cycle or on a path connecting two disjoint cycles. Let x be any vertex in S which is not on a cycle or a path connecting two disjoint cycles. Since G is connected and we have assumed that $N_p(G) \neq 0$, then by Corollary 17, G must contain a cycle C. Let P be a path from x to C. Let z be the vertex adjacent to x on P (cf. Fig. 8). Note that the edge xz is a bridge. Consider the connected component G_x containing x in $G-xz$. It follows that G_x is a tree (else x is on a path connecting two disjoint cycles) and therefore Algorithm N_p removes every vertex of G_x, which contradicts the assumption that x is in S. □

In order to illustrate the data structures which may be used to implement this algorithm, examples are shown in Fig. 9. For the given graph G, an array Degree, triply connected adjacency lists and a removal queue Q are shown.

Theorem 21. *Algorithm N_p can be implemented to run in $O(|E|+|V|)$ time.*

Proof. We can assume that the input to Algorithm N_p consists of a list of vertices and a list of edges (ordered pairs of vertices). The data structures in Fig. 9, consisting of a triply linked list of adjacencies and an array $(1 \ldots n)$ of vertex degrees, can be constructed in $O(|E|)$ time. In $O(|V|)$ time the vertices of degree at most 1 can be inserted into a simple queue Q. In $O(1)$ time Algorithm N_p can then remove a vertex of degree at most 1 from Q and update the linked lists, the array Degree, and the queue Q. Thus, the loop in Algorithm N_p can be executed in $O(|V|)$ time. □

References

[1] B. Bollobás, E.J. Cockayne and C.M. Mynhardt, On generalized minimal domination parameters for paths, Discrete Math. 86 (1990) 89–97.

[2] E.J. Cockayne, B.L. Hartnell, S.T. Hedetniemi and R. Laskar, Perfect domination in graphs, to appear in: J. Combin. Inform. Systems Sci.

[3] E.J. Cockayne, S.T. Hedetniemi and S. Mitchell, Linear algorithms on recursive representations of trees, J. Comput. Systems Sci. 18 (1979) 76–85.

[4] E.J. Cockayne and C.M. Mynhardt, On the product of k-minimal domination numbers of a graph and its complement, J. Comb. Math. Comb. Computing 8 (1990) 118–122.

[5] E.J. Cockayne, G. MacGillivray and C.M. Mynhardt, Generalized maximal independence parameters for paths and cycles, Quaestiones Math. 13 (1990) 123–139.

[6] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness. (Freeman, New York, 1979).

[7] E.O. Hare, S.T. Hedetniemi, R. Laskar, K. Peters and T.V. Wimer, Linear time computability of combinatorial problems on generalized-series-parallel graphs, Discrete Algorithms and Complexity, in: D.S. Johnson, ed., Proc. of the Japan–U.S. Joint Seminar, Perspectives in Computing, Vol. 15, (Academic Press, New York, 1987) 437–457.

[8] R.W. Irving, On approximating the minimum independent dominating set. Inform. Process. Lett. 37(4) (1991) 197–200.

[9] S.L. Mitchell, Algorithms on trees and maximal outerplanar graphs: design, complexity analysis and data structures study, Ph.D. Thesis, Dept. of Applied Mathematics and Computer Science, University of Virginia, 1977.

[10] T.V. Wimer, Linear algorithms on k-terminal graphs, Ph.D. Thesis, Dept. of Computer Science, Clemson University, 1987.

[11] T.V. Wimer, S.T. Hedetniemi and R. Laskar, A methodology for constructing linear graph algorithms. Congr. Numer. 50 (1985) 43–60.