R-Domination in Graphs

PETER J. SLATER

National Bureau of Standards, Washington, D C

ABSTRACT. The problem of finding a minimum k-basis of graph G is that of selecting as small a set B of vertices as possible such that every vertex of G is at distance k or less from some vertex in B Cockayne, Goodman, and Hedetniemi previously developed a linear algorithm to find a minimum 1-basis (a minimum dominating set) when G is a tree In this paper the k-basis problem is placed in a more general setting, and a linear algorithm is presented that solves the problem for any forest

KEY WORDS AND PHRASES computational complexity, dominating set, facility location, graph, k-basis, tree

CR CATEGORIES 5 32

1. *Introduction*

Given a finite, simple graph $G = (V, E)$ (that is, an undirected graph without loops or multiple edges), a subset D of the set V of vertices is called a *dominating set* [5] when every vertex not in D is adjacent to at least one vertex in D. More generally, if B is a subset of V then B will be called a *k-basis* ($k \geq 1$) when for each vertex v of V there is at least one vertex b of B such that the distance between b and v in G, denoted $d_G(b, v)$, is less than or equal to k. Thus a dominating set is a 1-basis.

While there are many applications of these ideas, an interpretation in terms of communication networks will be used here. If V represents a collection of cities and an edge represents a communication link, then, as in [3], one may be interested in selecting a minimum number of cities as sites for transmitting stations so that every city either contains a transmitter or can receive messages from at least one of the transmitting stations through the links. If only direct transmissions are acceptable, then one wishes to find a minimum 1-basis. If communication over paths of k links (but not of $k + 1$ links) is adequate in quality and rapidity, the problem becomes that of determining a *minimum k-basis*, i.e. a k-basis with the fewest possible vertices A *minimal k-basis* is a k-basis such that no proper subset of it is also a k-basis.

In [1] Cockayne, Goodman, and Hedetniemi present a (linear) algorithm for finding a minimum 1-basis of a tree (connected, acyclic graph). An abstract describing some unpublished work of Matula and Kolde concerning k-bases (and k-medians) of acyclic graphs appears in [4]. In the present paper a further generalization of k-bases to the concept of an R-dominating set is presented, and an algorithm that finds the minimum number of elements in an R-dominating set of a forest (an acyclic graph) is given. Letting p denote the number of vertices in V, an $O(p)$ (that is, linear) Fortran implementation of the algorithm is possible, as presented in the Appendix.

This work was done while the author was a National Academy of Sciences–National Research Council Postdoctoral Research Associate at the National Bureau of Standards, Washington, D C 20234.

Author's present address Applied Mathematics Division 5121, Sandia Laboratories, Albuquerque, NM 87115.

2. *R-Bases*

Suppose G has p vertices, and label these with the numbers 1 to p; that is, $V = \{1, 2, \cdots, p\}$. Suppose one is to select a collection B of vertices as sites for transmitting stations Rather than assuming that each vertex i must be within a distance of k from a vertex in B, suppose that to each i there corresponds a nonnegative integer a_i such that the distance from i to B must be at most a_i; that is, $d_G(i, B) \leq a_i$ where $d_G(i, B) = \min_{j \in B} d_G(i, j)$ (If $a_i = 0$, then vertex i must be in B.) One thinks of a_i as the "necessary distance (of vertex i) to a station," $NDTS(i) = a_i$. If subset B' of the vertices has been selected to be part of B, then the integers $b_i = d_G(i, B')$ are determined for all $i \in V$. One can think of b_i as the "distance (of vertex i) to an established station," $DTS(i) = b_i$.

Given graph G with $V = \{1, 2, \cdots, p\}$, suppose one has an ordered p-tuple of ordered pairs of integers, say $R = ((a_1, b_1), (a_2, b_2), \cdots, (a_p, b_p))$, where $a_i \geq 0$ and $b_i \geq 1$ for $1 \leq i \leq p$. Now $B \subseteq V$ will be said to *dominate* $i \in V$ if and only if (iff) either (1) there is a vertex b of B such that $d_G(i, b) \leq a_i$ or (2) there is a vertex j of V such that $d_G(i, j) + b_j \leq a_i$. If B dominates every vertex $i \in V$, then B will be said to be *R-dominating for G*. It will also be said that B *solves* (G, R). Note that the second condition is satisfied with $j = i$ when $b_i \leq a_i$, and if $b_i \leq a_i$ for every i then the null set is R-dominating. If condition (2) is met for vertex i, then an established station (at a distance of b_j from vertex j) is within distance a_i of vertex i. Viewed in this way, the "established stations" can be considered to exist external to V, as follows.

Given G and R, let $H = (V^*, E^*)$ be the graph obtained by adding to each vertex i a path of length b_i (see Figure 1), say $v(i, 0), v(i, 1), \cdots, v(i, b_i) = i$. For $0 \leq h \leq b_i - 1$, let $a_{v(i,h)} = b_{v(i,h)} = h$, and let E' denote the set of endpoints of H, namely $E' = \{v(1, 0), v(2, 0), \cdots, v(p, 0)\}$. Now B is R-dominating for G iff $B \subseteq V$ and given any $i \in V^*$ there is a vertex v of $B \cup E'$ such that $d_H(i, v) \leq a_i$. In Figure 1, vertex 4 of G is the only one for which $NDTS(i) > d(i, E')$. Thus either $\{4\}$ or $\{3\}$ is R-dominating

It is easy to prove that if $a_i < b_i$ for every vertex i of G, then B solves (G, R) iff for each vertex i of G condition (1) is solved, that is, $d_G(i, B) \leq a_i$. In particular, B is a k-basis for G iff B solves a (G, R) where R is of the form $((k, t_1), (k, t_2), \cdots, (k, t_p))$ with each $t_i \geq k + 1$.

A minimal R-dominating set is an R-dominating set B such that no proper subset of B is R-dominating. Let the *R-domination number of G*, denoted $\delta(G, R)$, be the smallest number of vertices in any minimal R-dominating set; such a set is called minimum or optimum. Since V itself solves (G, R) for any R, one always has $0 \leq \delta(G, R) \leq p$.

Suppose $R_1 = ((a_1, b_1), \cdots, (a_p, b_p))$ and $R_2 = ((c_1, d_1), \cdots, (c_p, d_p))$. One writes $R_1 \geq R_2$ to indicate that $a_i \leq c_i$ and $b_i \geq d_i$ for $1 \leq i \leq p$.

PROPOSITION 1. *If $R_1 \geq R_2$ and B solves (G, R_1), then B solves (G, R_2).*

PROOF. For condition (1) one has $c_i \geq a_i \geq d_G(i, b)$ for some $b \in B$, and for condition (2), one has $c_i \geq a_i \geq d_G(i, j) + b_j \geq d_G(i, j) + d_j$ for some $j \in V$.

COROLLARY 1.1. *If $R_1 \geq R_2$, then $\delta(G, R_1) \geq \delta(G, R_2)$.*

All of Ore's theorems [5, Ch. 13] concerning dominating sets can be generalized, as follows.

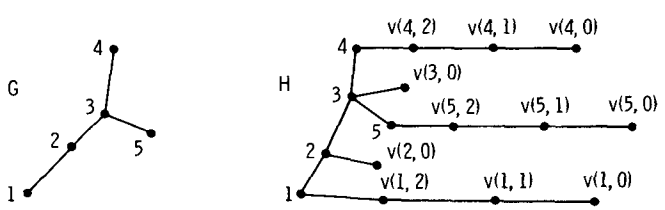

FIG 1 Forming H when $R = ((3,3), (2,1), (3,1), (1,3), (2,3))$

THEOREM 2. *Any R-dominating set contains a minimal one.*

THEOREM 3. *An R-dominating set B is a minimal R-dominating set iff for each vertex i of B there is a vertex i^* such that for any $j \in B - i$ one has $d_G(i^*, j) > a_{i^*}$ and for any h in V one has $d_G(i^*, h) + b_h > a_{i^*}$.*

THEOREM 4. *If G has no isolated vertices and $c_i \geq 1$ in $R = ((c_1, d_1), \cdots, (c_p, d_p))$ for every $i \in V$, then there exists an R-dominating set B such that $V - B$ is also R-dominating.*

PROOF. Let $R_1 = ((1, b), \cdots, (1, b))$ where $b \geq \max_{1 \leq i \leq p} d_i$ and $b \geq 2$. By [4, Th. 13.1.3] there is a set B such that B and $V - B$ are R_1-dominating. Now $R_1 \geq R$ implies that B and $V - B$ are also R-dominating.

Let the *R-domatic number of G*, denoted $d_R(G)$, be the maximum number of disjoint R-dominating sets [2]. Theorem 4 imples that if there is no vertex which must appear in every R-dominating set, then $d_R(G) \geq 2$.

THEOREM 5. *Suppose G has no isolated vertices and $a_i \geq 1$ in R for every vertex i. If B is a minimal R-dominating set, then $V - B$ is also an R-dominating set, in fact a 1-basis.*

PROOF. If $i \in V$, let $N(i)$ denote the set of all vertices in V which are adjacent to i. To show that $V - B$ is R-dominating and a 1-basis, it suffices to show that for each vertex i in B there is a vertex j in $N(i) \cap (V - B)$ (because each $a_i \geq 1$). Suppose to the contrary that $N(i) \subseteq B$, and let i^* be as in Theorem 3. First, $i^* \neq i$, for one can let j be any vertex in $N(i)$, which means that $j \in B - i$, and then $d_G(i^*, j) > a_{i^*} \geq 1 = d_G(i, j)$. Second, $i^* \notin N(i)$, for $d_G(i^*, i^*) = 0 \leq a_{i^*}$ implies $i^* \in B - i$. Thus $i^* \in V - (N(i) \cup \{i\})$. Now since B R-dominates G, since $j \in B - i$ implies $d_G(i^*, j) > a_{i^*}$, and since for any h in V one must have $d_G(i^*, h) + b_h > a_{i^*}$, it follows that $d_G(i^*, i) = d \leq a_{i^*}$. But a path from i^* to i of length d contains a path from i^* to some $j \in N(i) \subseteq B - i$ of length $d - 1 < a_{i^*}$. This contradiction shows that $i^* \notin V - (N(i) \cup \{i\})$. Thus $N(i) \not\subseteq B$.

THEOREM 6. *If G has no isolated vertices and $a_i \geq 1$ for every a_i in R, then for any minimal R-dominating set there is another disjoint from it. One therefore has $0 \leq \delta(G, R) \leq p/2$.*

3. *Statement of the Algorithm*

Algorithm RB, for determining a minimum R-dominating set of a forest, is given at the end of this section. The proof of its correctness is contained in the following lemmas. Proofs of the lemmas are relatively straightforward, and they are not presented here.

LEMMA 7. *If i is an isolated vertex of G and B is a minimal R-dominating set, then $i \in B$ iff $a_i < b_i$.*

LEMMA 8. *If $a_i = 0$, then B solves (G, R) iff $B = B' \cup \{i\}$ where B' is a solution to $(G - i, R')$, and $R' = ((a_1, b_1'), (a_2, b_2'), \cdots, (a_{i-1}, b_{i-1}'), (a_{i+1}, b_{i+1}'), \cdots, (a_p, b_p'))$ where $b_j' = b_j$ if j is not adjacent to i, and $b_j' = 1$ if j is adjacent to i.*

A vertex i is called *unicliqual* in G if the subgraph induced by $N(i)$ is complete; that is, any two vertices adjacent to i are also adjacent to each other. In particular, a vertex of degree one is unicliqual.

LEMMA 9. *If i is nonisolated and is unicliqual in G, then there exists an optimum solution B to (G, R) in which $i \notin B$, iff $a_i \neq 0$.*

For the following two lemmas it is assumed that vertex i has degree one, $a_i \geq 1$, and vertex j is adjacent to i.

LEMMA 10.1. *Assume $b_i > a_i$. A selection B of vertices with $B \subseteq V - i$ is a solution to (G, R) iff B is a solution to $(G - i, R')$ with $b_n' = b_n$ for every vertex n in $V - i$ and $a_n' = a_n$ when $n \neq j$ and $a_j' = \min(a_j, a_i - 1)$.*

LEMMA 10.2. *Assume $b_i \leq a_i$. A selection B of vertices with $B \subseteq V - i$ is a solution to (G, R) iff B is a solution to $(G - i, R')$ with $a_n' = a_n$ for every vertex n in $V - i$ and $b_n' = b_n$ when $n \neq j$ and $b_j' = \min(b_j, b_i + 1)$.*

The following algorithm can be used to find an optimum solution B to (G, R), that is, an R-dominating set B with $\delta(G, R)$ elements, if the removal of those vertices with $a_i = 0$ produces a forest. In particular, if G is a forest, then the algorithm can be used to find $\delta(G, R)$ for any R.

ALGORITHM RB

Step 1 Set $N = 0$ and $B = \varnothing$, the null set, and let H be the graph G

Step 2 Let i be a vertex of H with $a_i = 0$ If there are none, go to step 3 Increase N by one and put vertex i in B For each vertex j which is adjacent to i, make $b_j = 1$ Let $H = H - i$ If H has no vertices, go to step 5, otherwise, go to step 2

Step 3 Let i be a vertex of H of degree zero If there are none, go to step 4 If $a_i < b_i$, then increase N by one and put vertex i in B Let $H = H - i$ If H has no vertices, go to step 5, otherwise, go to step 3

Step 4 Let i be a vertex of H of degree one (Note that at this point one has $a_i \geq 1$) Suppose that j is the vertex adjacent to i If $a_i < b_i$, then let a_j be the minimum of its present value and $a_i - 1$ If $b_i \leq a_i$, then let b_j be the minimum of its present value and $b_i + 1$ Let $H = H - i$, and go to step 2

Step 5 Stop An optimum R-dominating set is B with cardinality $\delta(G, R) = N$

In the above algorithm a vertex i is put into B only when a_i is, or has been reduced to, zero or when i is, or has become, an isolated vertex with $a_i < b_i$.

Step 2 is justified by Lemma 8. By assumption, repetition of step 2 will create a forest If $F = (V, E)$ is a forest with vertex set $V = \{1, 2, \cdots, p\}$, then V can be arranged as $(i_1, i_2, \cdots, i_p)$ such that in $G - \{i_1, \cdots, i_{k-1}\}$ vertex i_k has degree zero or one. Thus at least one of step 2, step 3, and step 4 will always be applicable until one has considered every vertex in H. Step 3 is justified by Lemma 7. Step 4 is justified by Lemmas 9, 10.1, and 10.2.

Since each vertex is considered exactly once, a linear implementation of Algorithm RB is possible. A Fortran implementation of the algorithm is contained in the Appendix.

If one is interested in finding a k-basis, one can simply choose $a_i = k$ and $b_i = 1 + k$

Appendix. A Fortran Implementation of Algorithm RB

In the Fortran implementation of Algorithm RB which follows, it will be assumed that graph G is a forest with vertex set $V = 1, 2, \cdots, p$. Furthermore, $(i_1, i_2, \cdots, i_p)$ will be a listing of V such that in $G_k = G - \{i_1, \cdots, i_{k-1}\}$ vertex i_k has degree zero or one If the degree of i_k is zero in G_k, let $J_k = i_k$; if the degree of i_k is one in G_k, let J_k be the vertex adjacent to i_k in G_k.

It is assumed that the program input consists of P (the number of vertices in G), R (the P ordered pairs (a_i, b_i) which give $NDTS(i)$ and $DTS(i)$), and the description of G in the form (i_1, j_1), (i_2, j_2), $\cdots$, (i_p, j_p). Several arrays of length P are created: $NDTS(I)$ and $DTS(I)$ are as previously described; $LIST1(I)$ and $LIST2(I)$ are the lists of vertices $(i_2, i_2, \cdots, i_p)$ and $(j_1, j_2, \cdots, j_p)$, respectively; and $SET(I)$ will be the list of those vertices selected for the minimum R-dominating set.

For the code that follows, it is assumed that $P \leq 100$, and some statements (such as FORMAT and WRITE statements) are suppressed. To the right of each statement is listed the maximum number of times it can be executed. Letting $N1$ be the number of connected components of G, one has $A4 + A5 = N1$ and $A1 + A2 + A3 + A4 + A5 = P$.

```
      PROGRAM  RB                                                    
      INTEGER  P, LIST1(100), LIST2(100), NDTS(100)                  
      INTEGER  DTS(100), BASIS, SET(100)                             
C                                                                    
C       Assume that all arrays are initialized to zero               
C                                                                    
C       Read in P, R, and the list of edges                          
C                                                                    
      READ P                                                         1
      DO 11    I = 1, P                                              P + 1
```

```
      READ M, N                                                    P
      NDTS(I) = M                                                  P
      DTS(I) = N                                                   P
11    CONTINUE                                                     P
      DO 21  I = 1, P                                              P + 1
      READ M, N                                                    P
      LIST1(I) = M                                                 P
      LIST2(I) = N                                                 P
21    CONTINUE                                                     P
      I = 0                                                        1
      BASIS = 0                                                    1
C
C       The remainder of the program works from the endpoint and adjacency lists and
C         handles each vertex as necessitated by Lemmas 7, 8, 9, 10 1, and 10 2
C
C
10    I = I + 1                                                    P + 1
      IF(I GT P)GO TO 100                                          P + 1
      IF(LIST1(I) EQ LIST2(I)) GO TO 500                           P
      N = LIST2(I)                                                 P - N1
      M = LIST1(I)                                                 P - N1
      IF(NDTS(M) NE 0) GO TO 601                                   P - N1
      BASIS = BASIS + 1                                            A1
      SET(BASIS) = LIST1(I)                                        A1
      DTS(N) = 1                                                   A1
      GO TO 10                                                     A1
601   IF(DTS(M) GT NDTS(M)) GO TO 701                              P - N1 - A1
      DTS(N) = MIN0(DTS(N),DTS(M) + 1)                             A2
      GO TO 10                                                     A2
701   NDTS(N) = MIN0(NDTS(N),NDTS(M) - 1)                          A3
      GO TO 10                                                     A3
500   IF((NDTS(M) LT DTS(M)) GO TO 600                             N1
      GO TO 10                                                     A4
600   BASIS = BASIS + 1                                            A5
      SET(BASIS) = M                                               A5
      GO TO 10                                                     A5
100   PRINT "RESULTS"                                              1
      STOP                                                         1
      END
```

The results are contained in $BASIS$, which contains the R-domination number of G, and in SET, which is a list of elements in one (not necessarily unique) minimum R-dominating set of G.

The maximum number of statement executions is $17P + 9 + 3A1 + 2A2 + 2A3 + A4 + 3A5 - 3N1 \leq 20P + 9$.

REFERENCES

1 COCKAYNE, E J , GOODMAN, S E , AND HEDETNIEMI, S T A linear algorithm for the domination number of a tree Submitted for publication

2 COCKAYNE, E J , AND HEDETNIEMI, S T Dominating partitions of graphs To appear

3 LIU, C *Introduction to Combinatorial Mathematics* McGraw-Hill, New York, 1968

4 MATULA, D W , AND KOLDE, R A Multiple facilities location-allocation in certain sparse networks (abstract) *SIAM Rev 14* (1972), 533–534

5 ORE, O *Theory of Graphs* Amer Math Soc Colloquium Pub , Vol XXXVIII, Amer Math Soc , Providence, Rhode Island, 1962

RECEIVED APRIL 1975, REVISED SEPTEMBER 1975