Emergency Response Sets in Graphs

Jean Blair, US Military Academy

Wayne Goddard, Sandra M. Hedetniemi, Stephen T. Hedetniemi
Clemson University

Fredrik Manne, University of Bergen

Douglas Rall, Furman University

Abstract

We introduce a k-response set as a set of vertices where responders can be placed so that given any set of k emergencies, these responders can respond, one per emergency, where each responder covers its own vertex and its neighbors. A weak k-response set does not have to worry about emergencies at the vertices of the set. We define R_k and r_k as the minimum cardinality of such sets. We provide bounds on these parameters and discuss connections with domination invariants. For example, for a graph G of order n and minimum degree at least 2, $R_2(G) \leq 2n/3$, while $r_2(G) \leq n/2$ provided G is also connected and not K_3. We also provide bounds for trees T of order n. We observe that there are for each k trees for which $r_k(T) \leq n/2$, but that the minimum $R_k(T)$ appears to grows with k; a novel computer algorithm is used to show that $R_3(T) > n/2$. As expected, these parameters are NP-hard to compute, and we provide a linear-time algorithm for trees for fixed k.

1 Introduction

There are many facility placement problems on graphs. A typical question is to place facilities such that every edge or vertex is within some distance of a facility. This can be viewed as if guards are placed to maintain surveillance over a network. Although these types of problems are well studied, there has been very little research on the dynamic problem: what if the guards have to move to respond to the emergencies. One area of research is that of ***security*** [5], where guards need to respond to a *sequence* of emergencies. Another is the area of ***alliances*** [7], where one needs to be able to respond to *any set* of emergencies.

Here we consider the setting where one has to handle a *limited number* of *simultaneous* emergencies. This can be thought of as having to respond, for example, to multiple fires or power outages simultaneously. More specifically, given a graph we place facilities (guards) at a subset of vertices where each facility can respond to an emergency at the vertex where it is placed or at a neighboring vertex. The object is then to place the minimum number of facilities such that it is possible to respond to any k simultaneous emergencies. Similarly we also consider the case where an emergency does not occur at a vertex where a facility has been placed.

1.1 Definitions

Given an undirected graph $G = (V, E)$, we denote by $N[x]$ the closed neighborhood of a vertex x; that is, x union its neighbors. Consider subsets A and R of V. We say that R ***can handle*** A if a set of responders stationed at the vertices of R can respond to simultaneous emergencies at each vertex of A. In this, a responder stationed at a vertex $u \in R$ ***responds*** by either staying put or moving to a neighbor of u—so that there is a responder at each vertex of A. We are allowed only one responder and only one emergency per vertex. The set R is a ***k-response set*** of G if R can handle all subsets $A \subseteq V$ of size at most k. We define $R_k(G)$ of a graph G as the minimum cardinality of a k-response set of G.

While this parameter is our focus, it is also reasonable to consider the situation where emergencies cannot occur at the vertices where the responders are stationed. We call this version a ***weak k-response set***, and let $r_k(G)$ denote the minimum cardinality of a weak k-response set.

Recall that a ***k-dominating set*** is a set D of vertices such that every vertex not in D is adjacent to at least k vertices in D. The minimum cardinality of a k-dominating set is denoted γ_k. There are obvious inequalities among these invariants: for any k,

$$\gamma = \gamma_1 \leq r_k \leq R_k \leq \gamma_k.$$

We explicitly assume that $n \geq k \geq 2$ throughout (since the case $k = 1$ is exactly a dominating set).

2 Basic Results

In the following we present several basic results on the size of $R_k(G)$ and $r_k(G)$ both for general graphs and for some of the standard graphs such as the complete graph, the complete bipartite graph, and the cycle.

It is clear that:

Observation 2.1 *For a graph G of order n and any $k \leq n$:*
(a) $R_k(G) \geq k$.
(b) $r_k(G) \geq \min(k, n/2)$.
(c) If G has no isolate then $r_k(G) \leq n-1$.

Indeed, it is not hard to show that the only isolate-free graph G with $r_k(G) = n-1$ is the star $K_{1,n-1}$.

For a set R and vertex $v \in R$, a ***private neighbor*** of v (with respect to R) is a vertex w such that $N[w] \cap R = \{v\}$. If $w \neq v$ then w is an ***external*** private neighbor.

Observation 2.2 *A set R is a [weak] 2-response set iff*
(a) R is dominating; and
(b) every vertex in R has at most one [external] private neighbor.

PROOF. Necessity: Clearly R must be dominating. If a vertex has two private neighbors, then simultaneous emergencies at these neighbors causes problems.

Sufficiency: For any two emergencies their dominators can respond unless the emergencies are only dominated by the same vertex. But from (b) this will not happen. QED

We calculate next the values of R_k and r_k for the standard graphs. The first observation is straight forward:

Observation 2.3 *For the complete graph K_n: $r_k(K_n) = k$ for $k \leq n/2$ and $r_k(K_n) = \lceil n/2 \rceil$ for $k > n/2$, while $R_k(K_n) = k$ for $k \leq n$.*

Observation 2.4 *For the complete bipartite graph $K(a,b)$ with $a \leq b$:*
i) for $k \leq a/2$, $r_k = R_k = 2k$;
ii) for $a/2 \leq k \leq a$, $r_k = R_k = a$;
iii) for $a < k \leq b$, $r_k = R_k = b$;
iv) for $k \geq b$, $r_k = b$ and $R_k = k$.

PROOF. *(i,ii)* We have $k \leq a$. There are two cases. Suppose both sides have at least k vertices without responders. Then to respond to k emergencies on one side, one needs k responders on the other side. Therefore there must be at least $2k$ responders. Suppose one side has less than k vertices without responders. Then consider emergencies at the unprotected vertices on that side: after the movement of responders to those vertices, that side is full. Hence, there are at least $\min(a, b) = a$ responders. That is, between the two cases, the total number of responders is at least $\min(a, 2k)$.

Now, placing k responders on each side clearly works, as does placing a responders on the smaller side. Hence $r_k = R_k = \min(a, 2k)$.

(iii,iv) We have $k > a$. If the bigger side has k unprotected vertices, then one cannot respond to emergencies at those vertices. So the number of unprotected vertices there is less than k, and after responding to emergencies at those vertices, the bigger side will be full. Hence, we need at least b responders. For a weak k-response set, clearly b responders on the bigger side will work.

For a k-response set, we clearly need at least k responders. If $k > b$, any placement of k responders works. QED

Observation 2.5 *For fixed k, the cycle and the path on n vertices both have (a) $R_k = kn/(k + 2) + O(1)$, and (b) $r_k = kn/(2k + 1) + O(1)$.*

Proof. (a) Consider a tract of $k + 2$ successive vertices. One needs at least k responders in the tract to handle k emergencies in the middle of the tract. This gives the lower bound.

For the upper bound, partition the path or cycle into tracts of size $k+2$, and for each tract place k responders in the middle.

(b) Consider a tract of $2k+1$ successive vertices. To handle k emergencies in the tract, endpoints excluded, one needs at least k responders on the tract. This gives the lower bound.

For the upper bound, partition the path or cycle into tracts of size $2k + 1$, and for each tract place k responders on a vertex cover of the tract. QED

3 Bounds

If a graph G has minimum degree $\delta = k-1$, then the best upper bound for $R_k(G)$ is $n-(k-1)$ for large order n, which is achieved by the complete bipartite graph $K_{k-1,n-k+1}$. But if every vertex has at least k neighbors, there is an upper bound in term of n and δ. This follows immediately from the upper bound for k-domination: Cockayne, Gamble and Shepherd [2] showed that $\gamma_k(G) \leq kn/(k+1)$ for minimum degree at least k.

Observation 3.1 *For a graph G with $\delta(G) \geq k$, $R_k(G) \leq kn/(k+1)$, and this bound is sharp.*

A corresponding connected graph with $R_k(G) = kn/(k+1)$ can be found by taking a collection of disjoint cliques K_{k+1}, marking one vertex in each clique, and adding edges to turn the marked vertices into a clique.

The situation with weak response sets is different if the graph is required to be connected:

Theorem 3.1 *If $G \neq K_3$ is connected with $\delta(G) \geq 2$, then $r_2(G) \leq n/2$.*

Proof. The proof is by induction on n. The base case is $n = 4$: then G contains a spanning 4-cycle and $r_2(G) = 2$. Define a vertex as *small* if it has degree 2; and *large* otherwise.

We may remove edges while the graph—or rather its components—still obeys the hypothesis. So we may assume that no edge joins two large vertices unless its removal would create a K_3-component. Define P as the set of vertices having degree 3 with one large neighbor and two small neighbors that are adjacent. Let L be the large vertices except for P. We may assume there is no P–P edge: else we have a 6-vertex graph, and the theorem is easily checked for that graph.

Case 1: *Suppose there is an induced path a, b, c, d such that b and c are small.* Then let G' be the graph with $\{b, c\}$ contracted out; that is, delete b and c and add the edge a–d. If this graph satisfies the hypothesis, then let S' be a 2-response set of G'. This set can be extended to a 2-response set of G by adding one of $\{b, c\}$: if neither a nor d is in S', or if both a and d are in S', then either b or c will do; if only one of a and d is in S', then add the non-neighbor. If G' does not satisfy the hypothesis, then $G' = K_3$ and so $G = C_5$ and the result is easily checked.

Case 2: *There is a triangle a, b, c, a such that b and c are small and a has degree at least* 4. Then let G' be the graph with $\{b, c\}$ deleted. This graph satisfies the hypothesis and any 2-response set of G' can be extended to a 2-response set of G by adding either b or c.

Case 3: *There is no such path or triangle.* Then any small vertex either: (A) has two large neighbors; (B) is in a triangle (denoted a B-triangle) with another small vertex and a P-vertex; or (C) is in a 4-cycle (denoted a C-cycle) with two other small vertices.

Now, form a set S as follows. Take all of L (the large vertices, except for the P vertices). Then add one small vertex from each B-triangle, and the small vertex of each C-cycle that has two small neighbors. It is not hard to show that S is a 2-response set.

Let $|L| = l$ and let a denote the number of small vertices with two large neighbors. Let q denote the number of B-triangles and C-cycles combined. It follows that $n = l + a + 3q$ and $|S| = l + q$.

Now, by connectivity, $a \geq l - 1$. Further, let M be the sum of the degrees of the vertices in L. Then, trivially $M \geq 3l$, while $M \leq 2q + 2a$.

Hence, we need to maximize $l + q$ subject to the constraints $a \geq l - 1$, $a, l, q \geq 0$, $l + a + 3q = n$, and $3l \leq 2q + 2a$. This has a maximum of $(3n + 1)/7$, achieved at $q = (n + 5)/7$, $l = (2n - 4)/7$ and $a = (2n - 11)/7$. QED

Equality is achieved for the generalized corona: take disjoint C_4's, mark a vertex on each, and add edges to make the marked vertices connected. An example is given in Figure 1. (Note that this graph is not edge-minimal.)

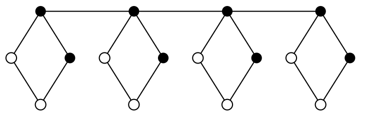


Figure 1: A connected graph with maximum r_2

This leaves open the question of a sharp upper bound for r_k for connected graphs.

3.1 Trees

We have already seen that the star has a large response value. We start with lower bounds for 2-response sets.

Theorem 3.2 *For any tree T,*
(a) $R_2(T) \geq 2(n + 1)/5$;
(b) $r_2(T) \geq (n + 1)/3$;
and these bounds are sharp.

Proof. (a) Consider any 2-response set R of cardinality d. For the forest induced by R, define a as the number of nontrivial components, and b the number of isolated vertices. Thus

$$d \geq 2a + b.$$

Define ℓ as the number of external private neighbors of vertices in R and $m = n - d - \ell$ as the number of remaining vertices outside R. By Observation 2.2, it follows that

$$\ell \leq d - b.$$

Further, we claim that

$$m \leq a + b - 1.$$

This can be seen by taking the tree T, discarding any edge both of whose ends are in $V - R$, then discarding some edges to make each vertex of $V - R$ have degree at most two, discarding any external private neighbors of R, and finally contracting each component of R to a single vertex. The result T' is a forest such that each of the m vertices has degree 2 and they form an independent set. That is, T' is formed from a forest with $a + b$ vertices by subdividing m different edges. That is, $m \leq a + b - 1$.

By straightforward algebra it then follows that $n \leq 5d/2 - 1$.

(2) The proof for a weak-response set R is similar, except that even an isolated vertex of R can have a private neighbor. In particular, we have:

$$\begin{aligned} &n = d + \ell + m, \\ &\ell \leq d, \\ &m \leq d - 1, \end{aligned}$$

whence the result. QED

For a tree which shows that the R_2 bound is sharp, start with the disjoint union of a P_4's. Then add $a - 1$ new vertices, each of degree 2, joined to a central vertex of two different P_4's such that the resultant graph on $5a - 1$ vertices is connected. A 2-response set of size $2a$ is given by taking the two central vertices on each of the P_4's. An example is the left tree of Figure 2.

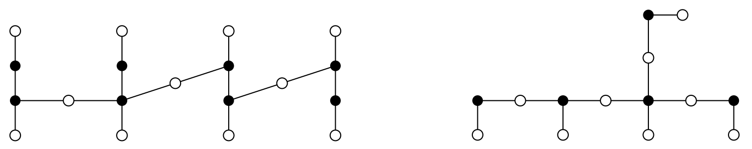


Figure 2: Trees with minimum R_2 and r_2

For a tree which shows that the r_2 bound is sharp, start with the disjoint union of d P_2's and mark one end of each. Then add $d - 1$ new vertices, each of degree 2, joined to the marked end of two different P_2's such that the resultant graph on $3d-1$ vertices is connected. A weak 2-response set of size d is given by taking the marked vertex on each P_2. An example is the right tree of Figure 2.

The question for a weak 3-response set is, surprisingly, almost the same.

Theorem 3.3 *For any tree T, $r_3(T) \geq (n+2)/3$, and this bound is sharp.*

Proof. Clearly $r_3(T) \geq r_2(T)$. In order to have a tree with $r_3(T) = (n+1)/3$, we need equality at several places in the proof of Theorem 3.2b. In particular, we need $\ell = d$ and $m = d-1$.

But that means that every vertex of R has an external private neighbor; in particular there is a pair P of vertices in R at distance 2. But one cannot handle an emergency at the private neighbors and the common neighbor of P simultaneously. QED

For a tree that shows this result is best possible, take a star with $d-1$ edges and subdivide each edge twice. The resultant octopus has $3d-2$ vertices and for a weak 3-response set of size d, take the central vertex and the middle vertex on each arm. An example is the left tree of Figure 3.

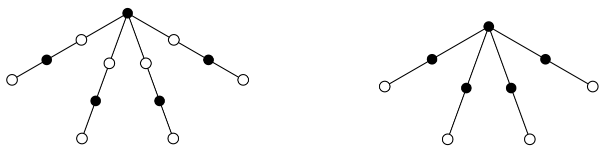


Figure 3: Trees with minimum r_3 and R_3

On the other hand,

Theorem 3.4 *For any tree T, $R_3(T) \geq (n+1)/2$, and this bound is sharp.*

This is proven in Section 5. An example of equality is the star with every edge subdivided once. A 3-response set is responders at every non-leaf vertex. An example is the right tree of Figure 3.

In general, the lower bound for $r_k(T)$ never grows beyond $n/2$, since any tree with a perfect matching has $r_k(T) \leq n/2$ for all k. However, the lower bound for $R_k(T)$ seems to grow as k increases.

4 Algorithmic Issues

We show that, as expected, the associated decision problems are NP-complete. We then determine linear-time algorithms for trees.

4.1 NP-Completeness

Both the k-response and the weak k-response decision problems are NP-hard to compute. We give the details for the simplest case:

Observation 4.1 *The question of, given graph G and integer b is $R_2(G) \leq b$, is NP-complete.*

Proof. Clearly the question is in NP. For the intractability we reduce from the domination number problem. Given a graph G, let H be the graph $2G + K_1$; that is, two disjoint copies G' and G'' of G together with a new vertex x adjacent to all other vertices.

We claim that $R_2(H) \leq 2k+1$ iff $\gamma(G) \leq k$. Let D be a dominating set of G of cardinality k. Then $D' \cup D'' \cup \{x\}$ is a 2-response set of H.

Conversely, let R be a 2-response set of H of cardinality $2k+1$. If $x \notin R$, then the restrictions of R to G' and G'' are both dominating sets, and one has cardinality at most k. If $x \in R$, then since x has at most one private neighbor, $R - \{x\}$ dominates all but at most one vertex y of H. It follows that the restrictions of $R - \{x\} \cup \{y\}$ to G and G' are both dominating sets, and the conclusion follows as before. QED

For example, this shows that the decision problem is NP-hard for chordal graphs (since the domination problem is hard for such graphs and the reduction preserves chordality).

4.2 Tree Algorithms

We next present an algorithm, for fixed k, for computing $R_k(T)$ on a tree T. We use the ideas pioneered in [8, 1]. Their work and subsequent work essentially shows that a linear-time algorithm exists; the task is to produce an explicit compact form for the algorithm. A similar approach can be used to produce an algorithm for computing $r_k(T)$, but we omit this construction.

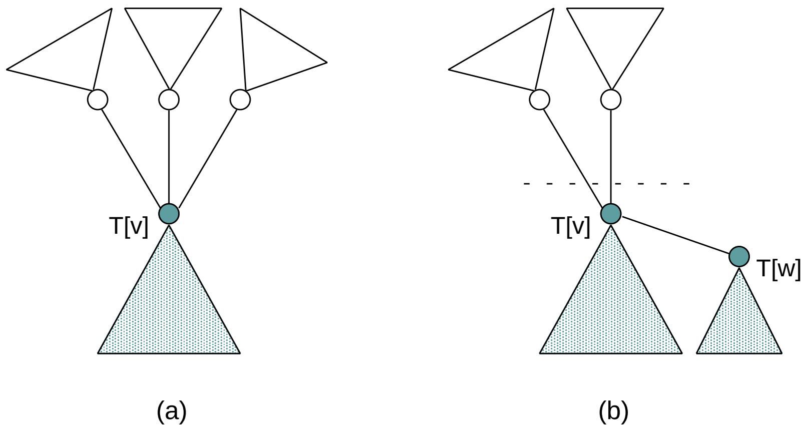


Figure 4: The tree algorithm.

Let $k \geq 1$ be fixed. Let S be a set of vertices in T where responders are placed. For $v \in S$, let r_v denote the responder placed at vertex v. When responding to an emergency, r_v can move to any vertex in $N[v]$.

Let $T[v]$ be a subtree of T containing vertex v and all of the vertices in zero or more of the subtrees in $T - \{v\}$. Figure 4a shows how $T[v]$ is constructed. The algorithm presented is based on first computing a solution for $T[v]$ and some subtree $T[w]$ such that $(v, w) \in E$ and $T[v] \cap T[w] = \emptyset$. We then show how to merge these results to form a combined solution for the subtree $T'[v] = \{T[v], T[w], (v, w)\}$ as shown in Figure 4b. In this way one inductively builds up solutions starting from leaves and single vertices and merge these until we have a solution for the whole tree.

The main idea of the algorithm is to consider the movement of responders between $T'[v]$ and $T - T'[v]$. The edges on which these responders can move are those that cross the dotted line in Figure 4b. Note that at most one responder can move down to v from $T - T'[v]$, while r_v can move up to at most one vertex in $T - T'[v]$. This yields four (disjoint) cases to be considered when T is handling a set of emergencies:

- $T'[v]$ *gets support:* Some $r_x \in T - T'[v]$ moves down to v.
- *No exchange:* No element in S moves between $T'[v]$ and $T - T'[v]$.
- $T'[v]$ *relays support:* $v \in S$ and r_v moves up to an adjacent vertex in $T - T'[v]$ while some $r_x \in T - T'[v]$ moves down to v.

- $T'[v]$ *gives support:* $v \in S$ and r_v moves up to an adjacent vertex in $T - T'[v]$ without some $r_x \in T - T'[v]$ moving down to v.

Consider a set $S \subseteq V$ that can handle any k emergencies. It follows that if $T'[v]$ gets support, then it *must* be able to handle $\min\{k, |T'[v]|\}$ emergencies. For the other three cases, the number of emergencies that $T'[v]$ can handle might be smaller than k. If $T'[v]$ relays support, then from the perspective of $T'[v]$, this is equivalent to v being in S and r_v being used only to handle an emergency at v. Thus when there is no exchange, $T'[v]$ can handle at least as many emergencies as when relaying support. Similarly, when it relays support, $T'[v]$ can handle at least as many emergencies as when giving support.

Given a rooted subtree U and a set S, we thus may define three values x, y and z with $x \geq y \geq z$:

- x is the maximum number of emergencies U can always handle when there is no exchange; (if $x \geq \min\{|U|, k\}$ we set $x = k$);
- y is the maximum number of emergencies U can always handle when relaying support; (if $y \geq \min\{|U|, k-1\}$ we set $y = k-1$; and if the root is not in S then we set $y = -1$);
- z is the maximum number of emergencies U can always handle when giving support; (same comments as with y).

(The reason for the bound of $k-1$ on y and z is that, if all k emergencies are in the subtree, then there is no need for relaying or giving support.) We say that the set S has property $\mathcal{H}[x, y, z]$ if given support it can handle $\min\{k, |U|\}$ emergencies and x, y and z are as above.

For example, if $k = 3$, U is the path on three vertices rooted at a leaf-vertex and S is the two leaf-vertices, then U has property $\mathcal{H}[2, 1, 0]$. For, it can handle two emergencies with no exchange, but not three; it can handle one emergency while relaying but not two; and the root is in S but cannot give support if there is an emergency at the root. More examples are given in Figure 5 later.

For a given set S for $T[v]$ and $T[w]$, we must determine first if the combined solution is feasible or not, and if it is, how many emergencies the combined solution can handle.

Lemma 4.1 *Consider a set S with properties $\mathcal{H}[a, b, c]$ in $T[v]$ and $\mathcal{H}[d, e, f]$ in $T[w]$. Then $T'[v]$ can handle $\min\{k, |T'[v]|\}$ emergencies while getting support if and only if $b + d + 1 \geq k$.*

Proof. Assume that $b + d + 1 \leq k - 1$. Consider $b + 1$ emergencies in $T[v]$ and $d + 1$ emergencies in $T[w]$. By the definition of d it follows that $T[w]$ must have support from $T[v]$ to handle this. In particular $b \geq 0$. When $T[v]$ is giving support to $T[w]$ it can only handle b emergencies even if it gets support from outside of $T[v]$. Since $b < k$, it follows that $T[v]$ cannot handle $b+1$ emergencies while giving support to $T[w]$. Hence, there is a set of $b + d + 2$ emergencies that $T'[v]$ cannot handle, even with support.

Assume that $b + d + 1 \geq k$. Then any d emergencies in $T[w]$ and any $k - d$ emergencies in $T[v]$ can be handled if $T'[v]$ gets support from $T - T'[v]$. If there are more than d emergencies in $T[w]$, then $T[w]$ needs support from $T[v]$. At the same time, there are at most $k-d-1$ emergencies in $T[v]$. But since $b+d+1 \geq k$ it follows that $b \geq k-d-1$ and $T[v]$ can supply $T[w]$ with the needed support. In the case that $b = -1$, we have $d = k$ and $T[w]$ does not need support. QED

If Lemma 4.1 does not apply, then the following result shows how to compute $\mathcal{H}[x, y, z]$ for $T'[v]$.

Lemma 4.2 *Consider a set S with properties $\mathcal{H}[a, b, c]$ in $T[v]$ and $\mathcal{H}[d, e, f]$ in $T[w]$. If $b + d + 1 \geq k$ then S has property $\mathcal{H}[x, y, z]$ in $T'[v]$ where*

$$\begin{aligned} x &= \min(a + f + 1, c + d + 1, k) \\ y &= \min(b, d, k - 1) \\ z &= \min(b, d, f + c + 1, k - 1). \end{aligned}$$

Proof. We consider the values of x, y, and z separately.

(x) In this case $T'[v]$ is not getting help from the outside. If $T[v]$ has $a + 1$ emergencies, then $T[v]$ needs support from $T[w]$. But if at the same time $T[w]$ has $f + 1$ emergencies, then $T[w]$ cannot give the needed support. It follows similarly that the tree cannot handle $c+1$ emergencies in $T[v]$ and $d+1$ in $T[w]$. Thus we must have $x \leq \min(a + f + 1, c + d + 1, k)$. We include the k here since there will never be more than k emergencies.

Now consider any $x = \min(a + f + 1, c + d + 1, k)$ emergencies where i emergencies are in $T[v]$ and j in $T[w]$. Then if both $i \leq a$ and $j \leq d$, the event can immediately be handled by the trees themselves. Next consider an event where $i \geq a + 1$ and therefore $j \leq f$. This can be handled by giving support from $T[w]$ to $T[v]$, and if $f = -1$ then $x \leq a$ and no support is needed. Similarly if $j \geq d + 1$, then $i \leq c$ and the event can be handled by giving support from $T[v]$

to $T[w]$. If $c = -1$ then $x \leq d$ and no support is needed. Thus it follows that $T'[v]$ can handle any combination of x emergencies and the result follows.

(y) Note first that if $v \notin S$ then $b = -1$, so that y will also be set to -1. If $v \in S$ and v is giving support to $T - T'[v]$ while at the same time getting support from $T - T'[v]$ then $T[v]$ can handle b emergencies. Since there cannot be any support given from $T[v]$ to $T[w]$, it follows that $T[w]$ can handle up to d emergencies. Thus the combined tree can handle the minimum of b and d emergencies and no more. If $T'[v]$ is giving support then there can at most be $k - 1$ emergencies in $T'[v]$; thus we include $k - 1$ in the formula.

(z) Again, note that if $v \notin S$ then $b = -1$, so that z will also be set to -1. If $v \in S$ and v is giving support to $T - T'[v]$ there can at most be $k - 1$ emergencies in $T'[v]$. There are three sets of emergencies that the combined tree cannot handle: Since $T[v]$ cannot give support to $T[w]$, $d+1$ emergencies in $T[w]$ cannot be handled. Also, if $T[v]$ gets support from w, it still cannot handle more than b emergencies. Finally, if there are at least $f + 1$ emergencies in $T[w]$, then w cannot give support to v, so that $T[v]$ can at most handle c emergencies. Note that this also covers the special case when $c = -1$. QED

Now, for a given rooted subtree U, let $C[x, y, z]$ be the minimum cardinality of an S with property $\mathcal{H}[x, y, z]$. If there is no such S, we set $C[x, y, z] = +\infty$. We compute $C[x, y, z]$ for $T'[v]$ and for all values of x, y, and z based on the values of $C[a, b, c]$ for $T[v]$ and all $k \geq a \geq b \geq c \geq -1$ and the values of $C[d, e, f]$ for $T[w]$ and all $k \geq d \geq e \geq f \geq -1$.

Consider first the case where U is a single vertex. Setting $S = V$ shows that $C[k, k-1, 0] = 1$ and setting $S = \emptyset$ shows that $C[0, -1, -1] = 0$. For all other values of x, y, and z, $C[x, y, z] = +\infty$ (meaning there is no such set).

Next consider the case where $T[v]$ and $T[w]$ are both non-empty. In order to compute all the C values for $T'[v]$, we must try every combination of $C[a, b, c]$ for $T[v]$ with every combination of $C[d, e, f]$ for $T[w]$ and, for each combination, test if $C[a, b, c] + C[d, e, f]$ is less than the current value of $C[x, y, z]$. If so, we store it. This complete scheme is given in Algorithm 1.

The minimum number of responders needed to handle any k emergencies in T is then given at the root by $C[k, y, z]$ for the smallest (lexicographic) pair $\langle y, z \rangle$ such that $C[k, y, z] \neq +\infty$. To compute their actual placement, it is sufficient to maintain pointers for each $C_v[x, y, z]$ to which solutions where used to create the solution.

Algorithm 1 An optimal k-emergency algorithm for trees.

```
procedure $k$-EMERGENCY($T = (V, E)$, $k$)
    Root $T$ in some vertex $r \in V$
    for each vertex $v \in V$ in postorder do
        $\forall a, b, c : C'_v[a, b, c] = +\infty$
        $C'_v[1, 1, 0] = 1$
        $C'_v[1, -1, -1] = 0$
        for each child $w$ of $v$ do
            for each $C_v[a, b, c] \neq +\infty$ do
                for each $C_w[d, e, f] \neq +\infty$ do
                    if $b + d + 1 < k$ then
                        $x = \min(a + f + 1, c + d + 1, k)$
                        $y = \min(b, d, k - 1)$
                        $z = \min(b, d, f + c + 1, k - 1)$
                        if $C_v[a, b, c] + C_w[d, e, f] < C'_v[x, y, z]$ then
                            $C'_v[x, y, z] = C_v[a, b, c] + C_w[d, e, f]$
        $C_v = C'_v$
```

Since there might be $O(k^3)$ C-values for each vertex, it follows that the overall time complexity of Algorithm 1 is $O(nk^6)$. However, for fixed k this is still linear.

5 Proof of Theorem 3.4

The proof of Theorem 3.4 is a computer proof. It uses the ideas introduced in [4]. The idea is that the table for the dynamic programming algorithm for R_3 contains (almost) the complete information about the parameter. So one can use the table (at least in principle) to calculate the value of the parameter on all trees of order n. Since we are interested in the minimum value, one can summarize the values for each n. This summary can be calculated recursively, and eventually a pattern emerges, which can then be proven by induction to hold for all n.

5.1 The Table for the R_3 Algorithm

The first step is to determine which of the properties $\mathcal{H}[x, y, z]$ actually arise. This can be done by starting with the singleton tree with $S = \emptyset$ and with $S = V$. These two cases have properties $\mathcal{H}[0, -1, -1]$ and $\mathcal{H}[3, 2, 0]$ respectively.

Then repeatedly combine every property for $T[v]$ with every property for $T[w]$ using Lemma 4.2 (and transitive closure). We call the tree $T[v]$ the ***parent*** and tree $T[w]$ the ***child***, and the tree $T'[v]$ the ***composition*** of the two trees, denoted $comp(T[v], T[w])$.

Lemma 5.1 *The thirteen properties shown in Figure 5 are the ones that are actually reachable.*

Figure 5 shows the 13 properties for R_3, together with a representative tree for each. (They are given in the order they are generated.) The table shows which property results when combining the indicated parent (row) and child (column), where for ease of sight, we have replaced the property by its row/column number. A dash means that the combination is invalid: it has $b + d < k - 1$ and fails the hypothesis of Lemma 4.2. In a tree, a solid vertex indicates a member of S.

Now, a tree together with a subset S of its vertices can be thought of as a ***marked tree***. The table represents a partition $\mathcal{P} = \{P_0, P_1, \ldots, P_{13}\}$ of the set of rooted marked trees, where P_0 is the impossible class (the markings that no valid tree can contain). Let π_ℓ be the set of pairs (i, j) such that combining a parent tree of class P_i with a child tree of class P_j produces a tree of class P_ℓ.

For a given unmarked tree T, define the vector $\mathcal{E}(T) = (e_1, \ldots, e_{13})$ as follows. For each class P_i, examine all such markings of T and determine the smallest number of marked vertices for that class. For example, in our case, e_1 is the minimum number of guards in a set with property $\mathcal{H}[0, -1, -1]$.

As in Algorithm 1, this vector can be calculated recursively from the vectors of the parent and child trees (P and C). Specifically, if $T = comp(P, C)$ with $\mathcal{E}(P) = \vec{x}$ and $\mathcal{E}(C) = \vec{y}$, then

$$\mathcal{E}(T) = comp(\vec{x}, \vec{y}) := \left(\min_{(i,j)\in\pi_\ell} \; x_i + y_j \right)_{\ell=1}^{13} .$$

5.2 From Table to Bound: Theory

Now we want the minimum value of the parameter. For this, we need only consider the ***set*** of all vectors for a given order:

$$\mathcal{E}(n) = \{\, \vec{x} : \vec{x} = \mathcal{E}(T) \text{ for some } T \text{ with } n \text{ vertices} \,\}.$$

This set too can be determined recursively:

$$\mathcal{E}(n) = \{\, comp(\vec{x}, \vec{y}) : 1 \le p < n,\ \vec{x} \in \mathcal{E}(p),\ \vec{y} \in \mathcal{E}(n-p) \,\}.$$

child

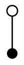
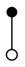
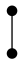
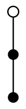
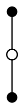
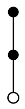
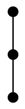
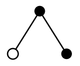
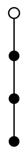
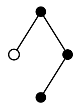
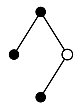
parent	$\mathcal{H}$[0,-1,-1] 1	$\mathcal{H}$[3,2,0] 2	 $\mathcal{H}$[1,-1,-1] 3	 $\mathcal{H}$[1,0,0] 4	 $\mathcal{H}$[3,2,1] 5	 $\mathcal{H}$[2,1,-1] 6	 $\mathcal{H}$[2,1,0] 7	 $\mathcal{H}$[2,1,1] 8	 $\mathcal{H}$[3,2,2] 9	 $\mathcal{H}$[2,0,0] 10	 $\mathcal{H}$[3,-1,-1] 11	 $\mathcal{H}$[3,0,0] 12	 $\mathcal{H}$[3,1,1] 13	
1	–	3	–	–	6	–	–	–	11	–	1	3	6	$\mathcal{H}$[0,-1,-1]
2	4	5	7	8	9	2	5	9	9	5	2	5	9	$\mathcal{H}$[3,2,0]
3	–	6	–	–	11	–	–	–	11	–	3	6	11	$\mathcal{H}$[1,-1,-1]
4	–	10	–	–	12	4	10	12	12	10	4	10	12	$\mathcal{H}$[1,0,0]
5	10	9	13	13	9	5	9	9	9	9	5	9	9	$\mathcal{H}$[3,2,1]
6	–	11	–	–	11	–	–	–	11	–	6	11	11	$\mathcal{H}$[2,1,-1]
7	–	13	7	8	13	7	13	13	13	13	7	13	13	$\mathcal{H}$[2,1,0]
8	–	13	8	13	13	8	13	13	13	13	8	13	13	$\mathcal{H}$[2,1,1]
9	12	9	13	13	9	9	9	9	9	9	9	9	9	$\mathcal{H}$[3,2,2]
10	–	12	–	–	12	10	12	12	12	12	10	12	12	$\mathcal{H}$[2,0,0]
11	–	11	–	–	11	–	–	–	11	–	11	11	11	$\mathcal{H}$[3,-1,-1]
12	–	12	–	–	12	12	12	12	12	12	12	12	12	$\mathcal{H}$[3,0,0]
13	–	13	13	13	13	13	13	13	13	13	13	13	13	$\mathcal{H}$[3,1,1]

Figure 5: The Table for R_3

To get it all started, one needs to know $\mathcal{E}(1) = \{\mathcal{E}(K_1)\}$. In our case, $\mathcal{E}(K_1)$ is $(0, 1, \sim, \ldots, \sim)$ (where $\sim$ means that there is no valid marking in this class).

Define $m(n)$ as the minimum of $R_3(T)$ taken over all trees T of order n. Then this is given by:

$$m(n) = \min \left\{ \min_{i \in \pi}(\vec{z}) : \vec{z} \in \mathcal{E}(n) \right\},$$

where π is the set of classes which correspond to a valid marking (all classes of the form $\mathcal{H}(3, y, z)$).

To determine the function $m(n)$, it is sufficient to replace $\mathcal{E}(n)$ with $\widetilde{\mathcal{E}}(n)$ where:

> $\widetilde{\mathcal{E}}(n)$ *is the minimal elements of the set* $\mathcal{E}(n)$*.*

Note that $\widetilde{\mathcal{E}}(n)$ can be calculated by taking the minimal elements in the set $\{\, comp(\vec{x}, \vec{y}) : 1 \le p < n,\ \vec{x} \in \widetilde{\mathcal{E}}(p),\ \vec{y} \in \widetilde{\mathcal{E}}(n-p) \,\}$.

5.3 Actual Calculations

We calculated $\widetilde{\mathcal{E}}(n)$ by computer and waited for a pattern. It turns out that $\widetilde{\mathcal{E}}(43)$ and $\widetilde{\mathcal{E}}(45)$ both have 54 vectors and are related by a ***simple shift***: that is, there is a bijection ϕ between the sets such that $\vec{x}$ and $\phi(\vec{x})$ have $\sim$ in the same components. Also $\widetilde{\mathcal{E}}(44)$ and $\widetilde{\mathcal{E}}(46)$ both have 43 vectors and are related by a simple shift. Further, the "same" simple shift that relates $\widetilde{\mathcal{E}}(43)$ and $\widetilde{\mathcal{E}}(45)$ also relates $\widetilde{\mathcal{E}}(45)$ and $\widetilde{\mathcal{E}}(47)$, and so on.

The proof of Theorem 3.4 is then a proof by induction that this simple shift pattern continues forever:

Lemma 5.2 *(a)* $m(n) = \lceil (n+1)/2 \rceil$ *for* $n < 43$ *except for* $m(3) = 3$.
(b) For $n \ge 43$*,* $\widetilde{\mathcal{E}}(n)$ *is given by the list given in Figure 6 (by adding* $\lceil n/2 \rceil$ *to every entry in every vector which is not* $\sim$*).*

The claim is that the set $\widetilde{\mathcal{E}}(n)$ is given by ***special cases*** for $n \le 42$ and a ***general form*** for $n \ge 43$. The proof is by induction. The proof entails computationally verifying the result by looking at all combinations of parent and child. In particular, we need to consider: parent special and child general form; child special and parent general form; and both parent and child general form.

This requires some computer algebra. In general, one needs to show that any vector α produced by the join of parent and child is in the ***envelope*** given by the

```
                         min over pi
~.~.~.~.~.~.0.0.2.~.2.3.0.  > 0
~.~.~.~.~.~.0.0.3.~.2.2.0.  > 0
~.~.~.~.~.1.0.0.2.~.1.1.1.  > 1
~.~.~.~.~.1.1.0.2.~.1.~.0.  > 0
~.~.~.~.~.2.0.0.1.~.1.2.1.  > 1
~.~.~.~.~.2.1.0.3.~.2.2.0.  > 0
~.~.~.~.~.3.1.0.2.~.2.3.0.  > 0
~.~.~.0.1.~.~.~.1.0.2.0.3.  > 0
~.~.0.0.1.0.~.~.1.0.1.1.2.  > 1
~.~.0.0.1.1.~.~.2.1.1.1.1.  > 1
~.~.1.0.1.1.~.~.1.0.1.1.1.  > 1
~.~.1.0.1.1.~.~.1.1.1.0.1.  > 0
~.~.1.1.2.~.~.~.1.1.0.0.~.  > 0
~.~.1.1.2.0.~.~.1.0.1.1.1.  > 1
~.~.1.1.2.1.~.~.1.0.1.0.~.  > 0
~.~.1.1.2.2.~.~.1.2.0.0.2.  > 0
~.~.2.2.3.2.~.~.1.0.0.0.~.  > 0
~.~.3.1.2.3.~.~.1.0.1.0.3.  > 0
~.1.~.1.1.~.1.2.1.1.1.1.1.  > 1
0.1.0.~.1.0.2.0.1.~.1.~.~.  > 1
0.1.0.0.1.0.0.2.1.~.1.~.~.  > 1
0.1.0.0.1.1.1.1.2.~.1.~.~.  > 1
0.1.0.1.1.0.1.1.1.~.1.~.~.  > 1
0.1.0.1.1.1.1.1.1.~.0.~.~.  > 0
0.1.0.1.1.1.1.2.2.1.1.2.1.  > 1
0.1.0.2.1.0.2.2.1.2.1.3.2.  > 1
0.1.1.1.2.0.2.~.1.~.0.~.~.  > 0
0.1.1.3.2.0.3.3.1.~.0.~.~.  > 0
1.0.1.1.2.1.~.~.1.~.0.~.~.  > 0
1.0.1.1.2.2.2.2.1.~.0.~.~.  > 0
1.0.2.2.3.0.3.~.1.~.0.~.~.  > 0
1.1.1.1.1.1.1.1.1.1.1.2.1.  > 1
1.1.1.1.2.0.1.~.1.~.0.~.~.  > 0
1.1.1.1.2.1.1.2.1.2.1.1.1.  > 1
1.2.0.~.1.0.~.2.1.~.0.~.~.  > 0
1.2.0.2.1.1.2.1.2.1.1.2.1.  > 1
1.2.0.3.1.0.2.1.1.2.1.3.1.  > 1
1.2.1.~.2.1.3.1.2.~.1.~.0.  > 0
1.2.1.1.2.1.0.2.2.1.1.1.1.  > 1
1.2.1.1.2.1.1.0.2.1.1.1.1.  > 1
1.2.1.2.1.1.2.3.1.1.2.1.1.  > 1
1.2.1.2.2.0.2.3.1.2.0.2.2.  > 0
1.2.1.3.1.0.3.3.1.2.0.2.2.  > 0
1.2.2.2.1.0.2.3.1.1.1.1.1.  > 1
2.0.2.2.3.0.2.~.1.~.0.~.~.  > 0
2.1.2.0.1.1.0.2.1.2.1.1.1.  > 1
2.1.2.0.1.1.1.1.2.2.1.1.2.  > 1
2.1.2.1.1.2.1.1.1.1.2.1.1.  > 1
2.1.2.2.1.1.2.0.1.2.1.2.1.  > 1
2.2.2.2.1.1.2.2.1.1.1.1.1.  > 1
2.3.1.3.2.1.3.2.1.2.0.3.1.  > 0
2.3.2.2.3.2.2.1.3.2.2.2.0.  > 0
3.0.3.1.2.2.2.2.1.3.1.2.3.  > 0
3.2.3.3.2.2.3.1.2.3.2.3.0.  > 0
```

```
                         min over pi
~.~.~.~.~.1.1.1.2.~.1.3.1.  > 1
~.~.~.~.~.1.1.1.2.~.2.2.2.  > 2
~.~.1.1.2.1.~.~.2.1.1.1.1.  > 1
~.~.2.0.1.2.~.~.2.2.1.1.3.  > 1
~.1.~.2.1.~.3.3.1.2.2.2.3.  > 1
~.2.~.1.2.~.1.3.2.1.2.1.1.  > 1
~.2.~.1.2.~.2.2.2.1.2.1.2.  > 1
0.1.0.2.1.2.2.2.2.~.1.~.~.  > 1
0.1.1.1.2.1.1.2.2.~.1.~.~.  > 1
1.1.1.1.1.1.1.1.2.~.1.~.~.  > 1
1.1.1.1.2.1.1.~.1.~.1.~.~.  > 1
1.2.~.~.2.0.~.~.1.~.2.~.~.  > 1
1.2.0.~.1.1.2.1.2.~.1.~.~.  > 1
1.2.1.~.1.1.~.~.1.~.2.~.~.  > 1
1.2.1.1.2.1.1.2.2.1.1.2.1.  > 1
1.2.1.2.2.1.2.1.2.2.2.2.2.  > 2
1.2.1.2.2.1.2.2.1.~.2.~.~.  > 1
1.2.1.3.2.1.3.1.2.3.1.3.1.  > 1
1.2.1.3.2.1.3.3.1.~.1.~.~.  > 1
1.2.2.~.1.1.~.~.1.~.1.~.~.  > 1
1.2.2.2.3.0.2.~.1.~.2.~.~.  > 1
2.1.2.1.2.1.1.2.2.3.1.2.2.  > 1
2.1.2.2.1.1.2.2.2.2.1.3.3.  > 1
2.2.2.1.2.2.1.1.2.1.2.2.1.  > 1
2.2.2.1.2.2.1.3.2.2.2.1.1.  > 1
2.2.2.1.2.2.2.2.2.2.2.1.2.  > 1
2.3.1.2.2.1.2.1.2.1.1.2.1.  > 1
2.3.2.~.1.0.~.~.1.~.3.~.~.  > 1
2.3.2.2.2.1.2.4.2.1.1.1.1.  > 1
2.3.2.2.2.1.3.3.2.1.1.1.2.  > 1
2.3.2.2.3.2.2.2.2.2.2.1.1.  > 1
2.3.2.3.3.0.3.3.1.~.3.~.~.  > 1
2.3.2.4.3.0.4.4.1.~.2.~.~.  > 1
2.3.3.~.1.0.~.~.1.~.2.~.~.  > 1
3.1.3.1.1.2.1.1.2.2.2.2.2.  > 1
3.1.3.1.2.2.1.~.1.3.2.2.2.  > 1
3.1.3.1.2.3.1.3.1.3.2.2.2.  > 1
3.2.2.3.1.2.2.1.2.2.1.3.2.  > 1
3.2.3.2.2.2.2.2.1.3.2.2.3.  > 1
3.2.3.2.3.2.2.4.1.4.1.2.3.  > 1
3.2.3.3.1.2.4.4.1.2.1.2.3.  > 1
3.4.3.3.4.1.3.3.2.3.1.1.1.  > 1
4.3.4.3.3.2.3.3.1.4.1.2.3.  > 1
```

Figure 6: $\widetilde{\mathcal{E}}(n)$ for n odd (left) and even (add $\lceil n/2 \rceil$ throughout)

claimed $\widetilde{\mathcal{E}}(n)$. One way to do this by computer is via linear programming, and show that there is some β in the claimed $\widetilde{\mathcal{E}}(n)$ which is component-wise better than α. At the same time, one needs to verify that every vector in the claimed $\widetilde{\mathcal{E}}(n)$ is in fact achievable. One way to do this by computer is to verify whether there is an integral solution of $\alpha = \beta$.

The software described in [4] produces the table, as shown in Figure 5, as well as the proof. The proof is, in principle, human checkable, but not in practice!

6 Other Questions

Grids. It can be shown for $P_m \,\square\, K_2$ that $R_2 = \lceil (4m+2)/5 \rceil$ and $r_2 = \lceil (2m+2)/3 \rceil$. But what is the behavior in general?

One can also consider multiple emergencies and multiple responders. One possibility is to allow double emergencies at the same vertex, but not allow double responders: this is equivalent to double domination [6] (not to be confused with 2-domination). Another possibility is to allow both double emergencies and double responders at the same vertex. This is a sort of Roman domination [3] extension.

References

[1] M. W. Bern, E. L. Lawler, and A. L. Wong. Linear-time computation of optimal subgraphs of decomposable graphs. *J. Algorithms*, 8(2):216–235, 1987.

[2] E. J. Cockayne, B. Gamble, and B. Shepherd. An upper bound for the k-domination number of a graph. *J. Graph Theory*, 9(4):533–534, 1985.

[3] Ernie J. Cockayne, Paul A. Dreyer, Jr., Sandra M. Hedetniemi, and Stephen T. Hedetniemi. Roman domination in graphs. *Discrete Math.*, 278(1-3):11–22, 2004.

[4] Wayne Goddard. Automated bounds on recursive structures. Submitted.

[5] Wayne Goddard, Sandra M. Hedetniemi, and Stephen T. Hedetniemi. Eternal security in graphs. *J. Combin. Math. Combin. Comput.*, 52:169–180, 2005.

[6] Frank Harary and Teresa W. Haynes. Double domination in graphs. *Ars Combin.*, 55:201–213, 2000.

[7] Petter Kristiansen, Sandra M. Hedetniemi, and Stephen T. Hedetniemi. Alliances in graphs. *J. Combin. Math. Combin. Comput.*, 48:157–177, 2004.

[8] T. V. Wimer and S. T. Hedetniemi. K-terminal recursive families of graphs. *Congr. Numer.*, 63:161–176, 1988. 250th Anniversary Conference on Graph Theory (Fort Wayne, IN, 1986).