The Firebreak Problem

Kathleen D. Barnetson,* Andrea C. Burgess,† Jessica Enright,‡
Jared Howell,§ David A. Pike,¶ Brady Ryan‖

May 13, 2020

Abstract

Suppose we have a network that is represented by a graph G. Potentially a fire (or other type of contagion) might erupt at some vertex of G. We are able to respond to this outbreak by establishing a firebreak at k other vertices of G, so that the fire cannot pass through these fortified vertices. The question that now arises is which k vertices will result in the greatest number of vertices being saved from the fire, assuming that the fire will spread to every vertex that is not fully behind the k vertices of the firebreak. This is the essence of the FIREBREAK decision problem, which is the focus of this paper. We establish that the problem is intractable on the class of split graphs as well as on the class of bipartite graphs, but can be solved in linear time when restricted to graphs having constant-bounded treewidth, or in polynomial time when restricted to intersection graphs. We also consider some closely related problems.

Key words: firebreak, separation, connectivity, computational complexity

AMS subject classifications: 05C85, 05C40, 68Q25

---

*Department of Mathematics and Statistics, Memorial University of Newfoundland, St. John's, NL, Canada.

†Department of Mathematics and Statistics, University of New Brunswick, Saint John, NB, Canada. *andrea.burgess@unb.ca*

‡School of Computing Science, University of Glasgow, Glasgow, Scotland. *Jessica.Enright@glasgow.ac.uk*

§School of Science and the Environment, Grenfell Campus, Memorial University of Newfoundland, Corner Brook, NL, Canada. *jahowell@grenfell.mun.ca*

¶Department of Mathematics and Statistics, Memorial University of Newfoundland, St. John's, NL, Canada. *dapike@mun.ca*

‖Department of Mathematics and Statistics, Memorial University of Newfoundland, St. John's, NL, Canada.

1 Introduction

In this paper we consider the FIREBREAK decision problem, which asks whether it is possible to establish a firebreak of a given size in a network represented by a graph and thereby protect a desired number of other vertices from being reached by a fire that breaks out at a specified vertex of the graph. The problem is formally stated as follows:

> FIREBREAK
>   Instance: A graph G, an integer k, an integer t, and a vertex $v_f \in V(G)$.
>   Question: Does $V(G)$ contain a k-subset S such that $v_f \notin S$ and the number
>   of vertices of $G - S$ that are separated from v_f is at least t?

There are similarities between this problem and the well-known FIREFIGHTING problem, which itself takes the form of a game with two players (fire and firefighters). The game begins with fire starting at a vertex. Thereafter, in each round of the game each firefighter is able to designate one unburnt vertex as a permanent firebreak, and then the fire spreads from each of its vertices to all of their unprotected neighbours. The game concludes when the fire can spread no more (which, in the case of infinite graphs, may result in a game that never terminates). Depending on context, the goal of the firefighters may be to minimise the number of vertices that are scorched, or to minimise the time in which the fire is contained. The FIREFIGHTING problem was introduced in 1995 by Hartnell [22] and has since attracted considerable attention. For a survey of results and open questions, see [18].

The FIREBREAK problem could be viewed as variant of the FIREFIGHTING problem in which the firefighters are active for only the initial round of the game, after which the fire spreads without further intervention. The particular nature and formulation of the FIREBREAK problem lends itself to several applications of practical interest. Although we model the problem in terms of fire, it readily applies to the spread of any contagion from a point of infection in a network and where a one-time response is able to be deployed in immediate reaction to the outbreak.

In the course of our investigation into the FIREBREAK problem, we noted that it is also closely related to what we will call the KEY PLAYER decision problem that pertains to the number of connected components that can be created by the removal of a set of vertices. By defining $c(G)$ to be the number of connected components of a graph G and $G - S$ to be the subgraph of G that is induced by the vertices of $V(G) \setminus S$, the KEY PLAYER problem can be formally described as follows:

> KEY PLAYER
>   Instance: A graph G, an integer k and an integer t.
>   Question: Does $V(G)$ contain a k-subset S such that $c(G - S) \geqslant t$?

The KEY PLAYER problem also models various real-world scenarios and applications of practical interest in networks. If we have the means to inoculate k nodes which then become impenetrable to the contagion, we can ask which k nodes to inoculate in order to create the greatest number of segregated quarantine zones. As another scenario, G might represent a communications network that we wish to disrupt by selectively disabling k of its nodes, with

the goal being to maximise the number of subnetworks that become unable to communicate with other subnetworks.

A related problem is one of dissemination rather than separation, whereby instead of selecting a k-set S so that $c(G - S)$ is maximised, we wish to selected a k-set S that can most efficiently reach the other vertices of G. One such scenario could be the spread of news: we wish to directly inform k individuals, who then propagate the news to their neighbours, so that the news spreads to everybody as quickly as possible. Instead of forwarding information, we might care about influence and opinion (such as might be the case in a marketing campaign, whereby k individuals are selected to receive a new commercial product in the hope that they will exert influence among their friends to acquire the product). This dissemination problem and its related separation problem appear to have been first jointly described by Borgatti in 2002 [12]. In [13] he refers to the dissemination scenario as the "Key Player Problem / Positive" and uses the phrase "Key Player Problem / Negative" for the problem involving the deletion of k nodes.

This "negative" variant corresponds to our interest. Some of the early papers about this problem in the literature attest to its applicability to network tolerance and robustness [5, 30, 34]). More recent results have considered it from a computational complexity perspective, establishing that it is NP-hard for various classes of graphs yet being solvable in polynomial time for graphs having bounded treewidth [1, 7, 28, 35]. A recent survey by Lalou *et al.* on the topic of detecting critical nodes in networks also touches on this problem [26]. Incidentally, our literature search revealed that there is also an edge-based version of the problem; although the edge version is beyond the scope of the present paper, we nevertheless provide a few references for the interested reader (see [6, 23, 31]).

In this paper we concentrate on the FIREBREAK decision problem, while also presenting some new results about the KEY PLAYER problem. In Section 2 we establish that both problems are NP-complete when restricted to the class of split graphs. We also find that the FIREBREAK problem is NP-complete when restricted to the class of bipartite graphs. Although the KEY PLAYER problem is NP-complete (and it remains so for planar cubic graphs), in the situation where the number of vertices to be removed from the graph coincides with the graph's connectivity the problem is found to be solvable in polynomial time. In Section 3 we consider graphs having treewidth that is bounded by a constant and for such graphs we show that the FIREBREAK problem can be solved in linear time, and in Section 4 we show that it can be solved in polynomial time for some classes of intersection graphs.

Before continuing, we establish some basic notation and terminology. For a graph $G = (V(G), E(G))$, we let $\deg_G(u)$ (or just $\deg(u)$ if the graph G is unambiguously implicit) denote the degree of vertex u, and we use the notation $u \sim v$ to indicate that vertex u is adjacent to vertex v. The maximum degree of G is $\Delta(G) = \max\{\deg(u) : u \in V(G)\}$. By $N_G(v)$ (or just $N(v)$ if there is no ambiguity) we denote the open neighbourhood of a vertex v, so that $N(v) = \{u \in V(G) : u \sim v\}$; the closed neighbourhood $N(v) \cup \{v\}$ will be denoted by $N[v]$. The *order* of a graph G is the cardinality $|V(G)|$ of its vertex set, and its *size* is the cardinality $|E(G)|$ of its edge set. If A is a subset of $V(G)$ then the subgraph of G induced by A, denoted $G[A]$, is the graph with vertex set A and edge set $E(G) \cap \big\{\{u, v\} : u, v \in A\big\}$. Throughout this paper we limit ourselves to finite undirected graphs without loops and without parallel edges, and so it follows that for the size of a graph G on n vertices we have $|E(G)| \leqslant \binom{n}{2} \in \mathcal{O}(n^2)$.

For an instance (G, k, t, v_f) of the FIREBREAK problem we define $\mathcal{F}(G, k, v_f)$ as the maximum number of vertices of $G - S$ that are not in the same connected component as v_f, where this maximum is taken over all choices for k-subsets $S \subseteq V(G) \setminus \{v_f\}$. Any k-subset $S \subseteq V(G) \setminus \{v_f\}$ that separates $\mathcal{F}(G, k, v_f)$ vertices from v_f will be called an *optimal* set.

2 Intractability

Clearly the FIREBREAK and KEY PLAYER decision problems are in NP since any given k-set S can be easily validated to determine whether S separates at least t vertices from v_f (for the FIREBREAK problem) or $c(G - S) \geqslant t$ (for the KEY PLAYER problem). In this section we show that these two decision problems are both NP-complete.

A *split* graph is any graph G that admits a vertex partition $V(G) = A \cup B$ such that $A \cap B = \emptyset$, $G[A]$ is a maximum clique, and B is an independent set (*i.e.*, B is a set of pairwise non-adjacent vertices). To prove that the FIREBREAK and KEY PLAYER decision problems are NP-complete, we will show that they are NP-complete even when the input graph G is restricted to the class of split graphs. We first note that under certain conditions the problems can be easily solved.

Lemma 1. *The FIREBREAK problem can be solved in linear time when $|N(v_f)| \leqslant k$.*

Proof. Let (G, k, t, v_f) be an instance of the FIREBREAK problem where G is a graph and $|N(v_f)| \leqslant k$. We now choose S to consist of every neighbour of v_f plus any choice of $k - |N(v_f)|$ additional vertices selected from $V(G) \setminus N[v_f]$. Clearly no vertex in $V(G) \setminus (S \cup \{v_f\})$ will be in the same connected component of $G - S$ as v_f. Since there are $|V(G)| - k - 1$ vertices in $G - (S \cup \{v_f\})$, the problem is equivalent to asking if $|V(G)| - k - 1 \geqslant t$. Hence one simply needs to count the vertices in G, which can be done in linear time. $\qquad \square$

Lemma 2. *The KEY PLAYER problem on a split graph G can be solved in linear time when k is at least the size $\omega(G)$ of a maximum clique in G.*

Proof. Let G be a split graph with $V(G) = A \cup B$, where A induces a maximum clique and B is an independent set, and let (G, k, t) constitute an instance of the KEY PLAYER problem. Observe that the maximum number of connected components will be produced by deleting A and $k - |A|$ of the $|V(G)| - |A|$ vertices of B. Thus the given instance of the KEY PLAYER problem has an affirmative answer if and only if $|V(G)| - k \geqslant t$. To solve this, one simply needs to count the vertices of G. This can clearly be done in linear time. $\qquad \square$

With the next result we show that when restricted to split graphs, the KEY PLAYER problem is no more difficult than the FIREBREAK problem.

Lemma 3. *The KEY PLAYER decision problem on split graphs can be solved in polynomial time with an oracle for the FIREBREAK decision problem on split graphs.*

Proof. Let (G_1, k_1, t_1) constitute an instance of the KEY PLAYER problem, where $G_1 = (A \cup B, E)$ is a split graph with a maximum clique on A and independent set B. Without

loss of generality we may assume that $k_1 < |A|$ for otherwise the problem is easily solved by Lemma 2.

We proceed to formulate an instance (G_2, k_2, t_2, v_f) of the FIREBREAK problem as follows. Construct G_2 from G_1 by adding a new vertex named v_f and adding an edge $\{u, v_f\}$ for each $u \in A$. Let $k_2 = k_1$ and $t_2 = t_1 - 1$.

Now suppose there exists a k_1-subset S_1 of $V(G_1)$ such that $c(G_1 - S_1) \geqslant t_1$. Since G_1 is a split graph, this means $G_1 - S_1$ must have at least $t_2 = t_1 - 1$ isolated vertices, none of which are in A. Thus we let $S_2 = S_1$. Then the t_2 isolated vertices of $G_1 - S_1$ are also isolated vertices of $G_2 - S_2$. As none of these vertices are members of the clique induced by A, none of them are in the same connected component as v_f. Thus S_2 is a k_2-subset of $V(G_2)$ such that $v_f \notin S_2$ and there are at least t_2 vertices in $G_2 - S_2$ not in the same component as v_f. Hence an affirmative answer to the KEY PLAYER problem implies an affirmative answer to the associated FIREBREAK problem (G_2, k_2, t_2, v_f).

Conversely, suppose that the FIREBREAK problem (G_2, k_2, t_2, v_f) has an affirmative answer, namely a k_2-subset S_2 of $V(G_2) \setminus \{v_f\}$ such that there are at least t_2 vertices of $G_2 - S_2$ that are not in the same connected component as v_f. Since v_f is part of the clique of G_2, these t_2 vertices must be isolated vertices in $G_2 - S_2$. Let $S_1 = S_2$. Then by the construction of G_2, these same t_2 vertices are also isolated in $G_1 - S_1$ and so they form t_2 distinct components in $G_1 - S_1$. Since by assumption $k_2 < |N(v_f)|$, then there must be at least one other vertex remaining in the clique with v_f. This vertex also remains in the clique of G_1, so $c(G_1 - S_1) = t_2 + 1 = t_1$ and hence the KEY PLAYER problem has an affirmative solution.

Since the instance (G_1, k_1, t_1) of the KEY PLAYER problem has an affirmative answer if and only if the associated instance (G_2, k_2, t_2, v_f) of the FIREBREAK problem has an affirmative answer, and this associated instance can clearly be constructed in polynomial time, then the KEY PLAYER problem can be solved in polynomial time with the availability of an oracle for the FIREBREAK problem. □

We now proceed to show that the FIREBREAK and KEY PLAYER decision problems are both NP-complete, even when restricted to the class of split graphs. To do so we will refer to the t-WAY VERTEX CUT problem studied by Berger *et al.* [7], expressed as a decision problem as follows:

t-WAY VERTEX CUT
    Instance: A graph G, an integer k and an integer t.
    Question: Does $V(G)$ contain a subset S such that $|S| \leqslant k$ and $c(G - S) \geqslant t$?

Theorem 1. *When restricted to split graphs, the FIREBREAK and KEY PLAYER decision problems are both NP-complete.*

Proof. Relying on a construction of Marx [28] that is restricted to split graphs, Berger *et al.* show that the t-WAY VERTEX CUT problem is NP-complete when restricted to split graphs [7]. By using an oracle for the KEY PLAYER problem, it is straightforward to answer any given instance (G, k, t) of the t-WAY VERTEX CUT problem. In particular, for each $k' \in \{1, 2, \ldots, k\}$ present the KEY PLAYER oracle with (G, k', t). The t-WAY VERTEX CUT problem has an affirmative answer if and only if one or more of the k answers provided by

the Key Player oracle is affirmative. Hence the Key Player problem is NP-complete when restricted to split graphs. It then follows from Lemma 3 that the Firebreak problem is also NP-complete when restricted to split graphs. □

Corollary 1. *The* Firebreak *and* Key Player *decision problems are* W[1]*-hard on split graphs with parameters k and t.*

Proof. For the Key Player problem, the result has already been proved by Theorem 16 of [28]. The reduction in Lemma 3 is clearly polynomial in k and t and is thus parameterized in k and t. Thus the Firebreak problem also is W[1]-hard on split graphs with parameters k and t. □

We can also use split graphs to show that the Firebreak problem is NP-complete for bipartite graphs.

Theorem 2. *When restricted to bipartite graphs, the* Firebreak *decision problem is* NP-*complete.*

Proof. Suppose (G, k, t, v_f) is an arbitrary instance of the Firebreak problem on a split graph G, where $V(G) = A \cup B$, A induces a maximum clique in G, and B is an independent set in G. Without loss of generality we may assume that $k < |N(v_f)|$, as otherwise the problem is easily solved by Lemma 1. We may further assume that $k < |A \cap N(v_f)|$ for otherwise an optimal set S can be obtained by selecting each vertex of $A \cap N(v_f)$ along with $k - |A \cap N(v_f)|$ vertices of $V(G) \setminus ((A \cap N(v_f)) \cup \{v_f\})$ such that vertices of $B \cap N(v_f)$ are preferentially selected prior to selecting any other vertices of $V(G) \setminus ((A \cap N(v_f)) \cup \{v_f\})$.

Let G' be the graph obtained from G by subdividing every edge of $G[A]$, so that $|V(G')| = |V(G)| + \binom{|A|}{2}$ and $|E(G')| = |E(G)| + \binom{|A|}{2}$. For convenience we let C denote the set $V(G') \setminus V(G)$ of $\binom{|A|}{2}$ vertices that are newly created in this process, and for any two distinct vertices $a_1, a_2 \in A$ let $c(a_1, a_2)$ be the vertex of C that is a common neighbour of a_1 and a_2. Observe that G' is a bipartite graph with vertex bipartition $(A, B \cup C)$. We will show that the instance (G, k, t, v_f) of the Firebreak problem has an affirmative answer if and only if the instance $(G', k, t + \binom{k}{2}, v_f)$ also has an affirmative answer.

First suppose that $S \subseteq (V(G) \setminus \{v_f\})$ is an optimal set for the instance (G, k, t, v_f). Since $k < |A \cap N_G(v_f)|$, in $G - S$ none of the vertices of $(A \setminus (S \cup \{v_f\}))$ are separated from v_f and so if there should exist some vertex $u \in S \cap B$ then for each $v \in A \setminus (S \cup \{v_f\})$ the k-set $(S \setminus \{u\}) \cup \{v\}$ is also optimal. From an iterated application of this observation it follows that there must exist an optimal set S_0 such that $S_0 \cap B = \emptyset$. In G', the number of vertices that are separated from v_f by S_0 is $\mathcal{F}(G, k, v_f) + \binom{|S_0 \cap A|}{2} = \mathcal{F}(G, k, v_f) + \binom{k}{2}$ and hence $\mathcal{F}(G', k, v_f) \geqslant \mathcal{F}(G, k, v_f) + \binom{k}{2}$.

Now let $S' \subseteq (V(G') \setminus \{v_f\})$ be an optimal set for the instance $(G', k, t + \binom{k}{2}, v_f)$ such that among all optimal sets, S' has the least intersection with C. Since $k < |A \cap N_G(v_f)|$, it follows that the set $A' = (A \setminus (S' \cup \{v_f\}))$ is not empty. Recall that $v_f \in V(G)$, so either $v_f \in A$ or $v_f \in B$.

Consider the situation in which $v_f \in B$. If there should exist a vertex $y \in A'$ that is separated from v_f in $G' - S'$, then it is necessary that $c(y, z) \in S'$ for each $z \in (A \cap N_G(v_f)) \setminus S'$

6

and consequently $k = |S'| \geqslant |A \cap N_G(v_f) \cap S'| + |(A \cap N_G(v_f)) \setminus S'|$. However, $k < |A \cap N_G(v_f)|$ and so none of the vertices of A' are separated from v_f in $G' - S'$.

If $v_f \in A$ then the set A' equals $(A \cap N_G(v_f)) \setminus S'$, which we partition into subsets Y and Z such that Y consists of all vertices of A' that are separated from v_f in $G' - S'$, and Z consists of all vertices of A' that are not separated from v_f in $G' - S'$. Since $k < |A \cap N_G(v_f)|$ then for some $a \in A'$ neither a nor $c(a, v_f)$ is in S', and hence $Z \neq \emptyset$. For each vertex $y \in Y$ it is necessary that $\bigcup_{z \in Z \cup \{v_f\}} \{c(y, z)\} \subseteq S'$. Consequently $k = |S'| \geqslant |A \cap N_G(v_f) \cap S'| + |Y|(1 + |Z|)$, which is at least $|A \cap N_G(v_f)|$ when $Y \neq \emptyset$ and thus it must be that $Y = \emptyset$. Therefore, regardless of whether v_f is in A or B, no vertex of A' is separated from v_f in $G' - S'$.

Let $w \in A'$. If there should exist some vertex $x \in S' \cap C$ then the k-set $(S' \setminus \{x\}) \cup \{w\}$ would contradict the selection of the set S' as an optimal set having minimum intersection with C. Hence $S' \cap C = \emptyset$. If there should exist some vertex $u \in S' \cap B$ then the k-set $(S' \setminus \{u\}) \cup \{w\}$ is also optimal. Iterated application of this observation ensures that there is an optimal set S'' such that $S'' \cap B = S'' \cap C = \emptyset$. In the graph G, the number of vertices that are separated from v_f by S'' is $\mathcal{F}(G', k, v_f) - \binom{|S'' \cap A|}{2} = \mathcal{F}(G', k, v_f) - \binom{k}{2}$ and hence $\mathcal{F}(G, k, v_f) \geqslant \mathcal{F}(G', k, v_f) - \binom{k}{2}$.

So when $k < |A \cap N_G(v_f)|$ we conclude that $\mathcal{F}(G', k, v_f) = \mathcal{F}(G, k, v_f) + \binom{k}{2}$. Thus the FIREBREAK instance (G, k, t, v_f) for the split graph G has an affirmative answer if and only if the instance $(G', k, t + \binom{k}{2}, v_f)$ also has an affirmative answer. $\square$

2.1   Some further comments about the KEY PLAYER problem

While the FIREBREAK problem is the main focus of this paper, we do have some additional results pertaining to the KEY PLAYER problem. In Theorem 1 it was established that the KEY PLAYER problem is not only NP-complete, but also that it remains so when restricted to the class of split graphs. It happens that the problem is also NP-complete when restricted to cubic planar graphs, in contrast to the FIREBREAK problem which we now show can be solved in polynomial time when restricted to graphs of constant-bounded degree (including cubic graphs).

Lemma 4. *The* FIREBREAK *problem can be solved in polynomial time on graphs of constant-bounded degree.*

Proof. Let m be a fixed integer and let (G, k, t, v_f) be an instance of the FIREBREAK problem where G is a graph on n vertices such that $\Delta(G) \leqslant m$. If $k \geqslant \deg(v_f)$ then the answer is affirmative if and only if $t \geqslant n - 1 - k$ because separating the maximum number of vertices from v_f is accomplished by deleting $N(v_f)$ plus $k - |N(v_f)|$ other vertices. If $k < \deg(v_f)$ then the answer can be computed by exhaustively considering all $\binom{n}{k}$ k-subsets of $V(G) \setminus \{v_f\}$ and determining whether any of them separate at least t vertices from v_f. Since k is bounded by the constant m, this computation can be done in polynomial time. $\square$

Showing that the KEY PLAYER problem is intractable on cubic planar graphs involves consideration of the well-known INDEPENDENT SET problem.

INDEPENDENT SET

    Instance: A graph G and an integer m.

    Question: Does G contain an independent set of at least m vertices?

Theorem 3. *The* KEY PLAYER *decision problem on cubic planar graphs is* NP-*complete.*

Proof. We employ a reduction from the INDEPENDENT SET problem, which Garey and Johnson established in 1977 to be NP-complete on 3-regular planar graphs [19]. Given an instance (G, m) of the INDEPENDENT SET problem, construct an instance $(G, |V(G)| - m, m)$ of the KEY PLAYER problem. It is easy to see that INDEPENDENT SET has an affirmative answer if and only if this instance of the KEY PLAYER problem has an affirmative answer. Thus an oracle for the KEY PLAYER problem can be used to efficiently solve the INDEPENDENT SET problem.   □

Having established that the KEY PLAYER problem is intractable for a variety of classes of graphs, we now proceed to consider the special case in which the parameter k is restricted to being the connectivity of the graph in question. For any nontrivial graph G, its connectivity $\kappa(G)$ is the size of a smallest set S of vertices such that $G - S$ is not a connected graph; such a set S is called a *cut* (or *k-cut* when we wish to explicitly mention the size of the cut).

As we shall see, when $k = \kappa(G)$ the KEY PLAYER problem can be solved in polynomial time. An initial thought for how to potentially prove that this is so is to enumerate all of the $\kappa(G)$-cuts of G and if they are polynomial in number then simply calculate $c(G - S)$ for each $\kappa(G)$-cut S. However, unlike edge cuts of size $\kappa'(G)$ (of which there are at most $\binom{n}{2}$; see Section 4.3 of [29] for a proof), there can be exponentially many $\kappa(G)$-cuts in a graph G, as is the case with the graph illustrated in Figure 2.1. Hence the naïve idea of examining each $\kappa(G)$-cut individually will not serve as a valid approach.
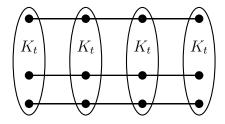

Figure 1: A graph G on $n = 4t$ vertices. G consists of four copies of K_t that are joined by t disjoint paths of length 3. The graph has connectivity $\kappa(G) = t$ and more than 2^t cut sets of size $\kappa(G)$.

Instead, we consider the notion of a *k-shredder* in a k-connected graph, which is defined to be a set of k vertices whose removal results in at least three components being disconnected from one another; note that it is necessary here that $k \geqslant \kappa(G)$ since G must be k-connected and each k-shredder is also a k-cut. An algorithm that is capable of finding all of the k-shredders of a graph G on n vertices in polynomial time is presented in [15]. The actual

number of k-shredders is determined in [17] to be at most $\frac{2n}{3}$ when $k \geqslant 4$. We can thus solve the KEY PLAYER problem in polynomial time when $k = \kappa(G)$ by following the steps of the algorithm presented below.

Algorithm 1.

1. Let $k = \kappa(G)$.

2. If $k \geqslant 4$ then use the algorithm of [15] to find all of the k-shredders of G. As [17] asserts that there are at most $\frac{2n}{3}$ of them, we then exhaustively check to see which k-shredder S maximises $c(G - S)$. If there should happen to be no k-shredders, then it must be that every k-cut produces exactly two components.

3. If $k \leqslant 3$ then exhaustively check each k-subset S of $V(G)$ to see which k-sets are cut sets, and then determine which k-cut maximises $c(G - S)$. The number of k-subsets that must be checked is $\binom{n}{k}$, which is polynomial in n since k is either 1, 2 or 3.

4. Having determined the maximum number of components that can result from the deletion of any k-cut, compare this quantity with t to answer the given instance of the KEY PLAYER problem.

3 Graphs with Constant-Bounded Treewidth

Before we review the technical details of treewidth, we first observe that the FIREBREAK problem can be easily solved in the case where the graph in question is a tree.

Theorem 4. *The* FIREBREAK *problem can be solved in polynomial time on trees.*

Proof. Suppose (T, k, t, v_f) is an instance of the FIREBREAK problem where T is a tree on n vertices. If $|N(v_f)| \leqslant k$ then a polynomial time solution follows from Lemma 1, so we henceforth assume that $|N(v_f)| > k$.

Root the tree T at v_f. For each vertex v of T define $T(v)$ to be the subtree of T rooted at v and let $\mathcal{T}(v)$ denote the number of vertices in $T(v)$. If v is a leaf in the tree then clearly $\mathcal{T}(v) = 1$. For any other vertex v with children $x_1, \ldots, x_m$, we have that $\mathcal{T}(v) = \mathcal{T}(x_1) + \cdots + \mathcal{T}(x_m) + 1$. The computation of each $\mathcal{T}(v)$ can be clearly done in $\mathcal{O}(n)$ time.

To find an optimal k-set S (namely one that separates the most vertices from v_f) simply select k vertices in $N(v_f)$ having the k greatest subtree sizes. If $v_1, \ldots, v_k$ are these k vertices, then there are $\mathcal{T}(v_1) + \cdots + \mathcal{T}(v_k) - k$ vertices not in the same connected component of v_f.

To answer the given instance of the FIREBREAK problem, it now suffices to ask if $\mathcal{T}(v_1) + \cdots + \mathcal{T}(v_k) - k \geqslant t$. Since both the computation of each $\mathcal{T}(v)$ and the identification of the k largest values of $\mathcal{T}(v)$ can be done in polynomial time, the problem can therefore be answered in polynomial time. $\qquad\square$

The proof of Theorem 4 comes close to proving that the FIREBREAK problem can be solved in linear $\mathcal{O}(n)$ time for a tree on n vertices, except that the task of selecting the k neighbours of v_f with the greatest subtree sizes may require a nonlinear sort to be performed. However, it will be shown later in this section that a linear time solution does nevertheless exist. Our approach will be to consider the effect that bounded treewidth has on the complexity of the problem. The treewidth parameter, defined below, was introduced by Robertson and Seymour [32].

Definition 1. A *tree decomposition* of a graph G is a pair $(X, T = (I, F))$ where $X = \{X_i : i \in I\}$ is a family of subsets of $V(G)$, and T is a tree whose vertices are the subsets X_i such that:

1. $\bigcup_{i \in I} X_i = V(G)$.

2. For every edge $uv \in E(G)$, both u and v are in some X_i, $i \in I$.

3. If i, j, k are vertices of T, and k lies on the (unique) path from i to j, then $X_i \cap X_j \subseteq X_k$.

The *width* of a tree decomposition is $\max\{|X_i| - 1 : i \in I\}$. The *treewidth* $\mathrm{tw}(G)$ of a graph G is the minimum width of all tree decompositions of G.

It is easy to see that trees have treewidth at most 1. Other graphs with small treewidth are, in a sense, tree-like. For instance, if the treewidth of a graph G is bounded by a constant (*i.e.*, $\mathrm{tw}(G) \leqslant w$), then it follows from Lemma 3.2 of [33] that $|E(G)| = \mathcal{O}(n)$ where $n = |V(G)|$. Graphs that are in some way similar to trees often lend themselves to tractable solutions for problems that are intractable for graphs in general (see [2, 4, 8] for details of several examples).

Moreover, Bodlaender has presented an algorithm that finds a tree decomposition of a graph G in time that is linear in the number of vertices and exponential in the cube of the treewidth [9]. For graphs having constant-bounded treewidth, it is therefore possible to find tree decompositions in linear time.

Theorem 4 demonstrated that the FIREBREAK problem is easily solved for trees. To show that it is also tractable for graphs for which the treewidth is bounded by a constant, we will rely on a powerful result that is based on work of Courcelle, independently proved by Borie, Parker and Tovey, and further extended by Arnborg, Lagergren and Seese [3, 14, 16]. A survey by Langer *et al.* [27] presents it in a slightly more general form than we require. For our purposes, the following will suffice:

Theorem 5 (see Theorem 30 of [27])**.** *Let G be a graph on n vertices, let w be a constant, and let $\mathcal{P}$ be a graph theoretic decision problem that can be expressed in the form of extended monadic second-order logic. If $\mathrm{tw}(G) \leqslant w$ then determining whether G has property $\mathcal{P}$ can be accomplished in time $\mathcal{O}(f_{\mathcal{P}}(w) \cdot n)$ where $f_{\mathcal{P}}$ is a function that depends on the property $\mathcal{P}$.*

Although the function $f_{\mathcal{P}}$ may not be polynomial, if w is a constant then so too is $f_{\mathcal{P}}(w)$. Hence decision problems that have extended monadic second-order (EMSO) formulations are fixed-parameter tractable. Monadic second-order (MSO) logic expressions for graphs are based on

- variables for vertices, edges, sets of vertices and sets of edges,
- universal and existential quantifiers,
- logical connectives of conjunction, disjunction and negation,
- and binary relations to assess set membership, adjacency of vertices, incidence of edges and vertices, and equality for vertices, edges and sets.

We will only need to consider vertices and sets thereof, which will be respectively denoted by lower case and upper case variable names. Predicates can be constructed from the basic ones and incorporated into expressions (in this manner a predicate for implication can be built). To provide an illustrative example, the expression

$$\exists X \exists Y \left(\forall u \left((u \in X) \wedge (u \notin Y) \right) \vee \left((u \notin X) \wedge (u \in Y) \right) \right) \wedge \\ \left(\forall u \forall v \left((\text{adj}(u,v)) \Rightarrow ((u \in X) \wedge (v \in Y)) \vee ((u \in Y) \wedge (v \in X)) \right) \right)$$

encodes whether a given graph is bipartite, where $\text{adj}(u,v)$ represents a Boolean predicate that evaluates whether vertices u and v are adjacent.

Extended MSO logic has additional features that enable set cardinalities to be considered. The survey by Langer *et al.* [27] provides an excellent overview, to which we direct readers for more details.

Since the factor $f_{\mathcal{P}}(w)$ in the conclusion of Theorem 5 effectively becomes a hidden constant, it follows that deciding whether a graph G has the property $\mathcal{P}$ can be done in linear time when the hypothesis of the theorem is satisfied. With this in mind, we now show that the FIREBREAK problem is tractable when restricted to graphs having treewidth at most a constant w.

Theorem 6. *The* FIREBREAK *problem can be solved in linear time for graphs with constant-bounded treewidth.*

Proof. Let φ represent the following logic expression with two set variables (S and X).

$$\varphi = (v_f \notin S) \wedge (v_f \notin X) \wedge \left(\forall y \, (y \in S) \Rightarrow (y \notin X) \right) \wedge \\ \left(\forall x \forall y \left((x \in X) \wedge (\text{adj}(x,y)) \wedge (y \notin S) \right) \Rightarrow (y \in X) \right)$$

Observe that φ encodes whether the set S separates the set X from a designated vertex v_f. To take into consideration the cardinalities of the sets S and X, we now describe how to construct an evaluation relation ψ as indicated by Definition 18 of [27]. Following the notation of [27], given that we have two sets (S and X) and two integer input values (k and t), choosing $m = 1$ will result in ψ having four variables:

$$y_1 = \sum_{u \in S} w_1(u) \qquad y_2 = \sum_{u \in X} w_1(u) \qquad y_3 = k \qquad y_4 = t$$

Define the weight function $w_1 : V(G) \to \mathbb{R}$ such that $w_1(v) = 1$ for each $v \in V(G)$. Now, let ψ be the evaluation relation

$$(y_1 = y_3) \wedge (y_2 \geqslant y_4)$$

We have adhered to Definition 18 of [27]. Hence we have created an EMSO expression that encodes the FIREBREAK decision problem. The result now follows from Theorem 5. ☐

4   Intersection Graphs

Given a family of sets $\mathcal{S} = \{S_0, S_1, \ldots, S_k\}$, the *intersection graph* of $\mathcal{S}$ is a graph $G = (V, E)$ for which there exists a bijection f between V and $\mathcal{S}$ such that u is adjacent to v in G if and only if $f(u) \cap f(v) \neq \emptyset$, that is, $f(u)$ *intersects* $f(v)$. We say that the bijective assignment and the family of sets are a *representation* of G. When we restrict the nature of the representing sets, we can restrict the class of representable graphs, and structured representations have provided a wide variety of tractability results (many examples are listed in [20]).

Here, we give polynomial-time algorithms for two classes of intersection graphs: subtree intersection graphs of limited leafage and permutation graphs. In both cases, we use an approach that sweeps the representation for separators, allowing us to exhaustively check these separators for firebreak feasibility.

4.1   Intersection graphs of subtrees in a tree

In this section we focus on the intersection graphs of subtrees in a tree of constant bounded leafage, for which we show that the FIREBREAK problem can be solved in polynomial time. For our purposes, two subtrees are considered to intersect if they share at least one vertex. The intersection graphs of a tree are the chordal graphs, and the intersection graphs of trees with a constant bounded number of leaves (the *leafage*) can be recognised and a representation constructed in time $\mathcal{O}(n^3)$ [21] (where n is the number of vertices in the graph to be represented). Because these are a subfamily of chordal graphs, for which $\mathrm{tw}(G) = \omega(G) - 1$, they do not in general have constant-bounded treewidth and so the results from Section 3 do not apply to them.

Theorem 7. *The* FIREBREAK *problem on a graph* $G = (V, E)$ *that is the intersection graph of subtrees of a tree of leafage* ℓ *can be solved in time* $\mathcal{O}(|V|^{2\ell+1})$.

Proof. Let (G, k, t, v_f) be an instance of the FIREBREAK problem where G is the intersection graph of subtrees $\mathcal{T}$ of a tree $T = (V_T, E_T)$ with constant bounded number of leaves ℓ, and denote by $\mathcal{T}(v)$ the subtree of T that represents vertex $v \in V(G)$. For convenience, let $n = |V|$. Recall that we denote by S the set of k vertices that we remove from the graph G in order to form a firebreak.

We make a simplifying assumption that we should not place in S a neighbour of v_f that is adjacent only to other neighbours of v_f, as such a vertex being included in S can only ever protect from burning that single vertex and no others; there is always a non-worse choice of vertex for inclusion in S. Note also that we assume that v_f has more than k neighbours, for if not then we apply Lemma 1 to resolve the question in $\mathcal{O}(n)$ time.

We argue that we can find a polynomially-bounded number of useful minimal separators, that any solution to the FIREBREAK problem will place in S at most a constant bounded number of them, and that we can check each candidate set of separators efficiently.

Given the representation $\mathcal{T}$, which by [21] we can construct in time $\mathcal{O}(n^3)$, we know from [25] that there are at most $\mathcal{O}(n^2)$ minimal vertex separators in our graph G, and that they correspond to the vertices of T: specifically, there is one for each vertex u of T, and it is composed of the vertices of G that are represented by subtrees that contain u. Note that (also due to [25]) there are at most $\mathcal{O}(n^2)$ vertices in T. Any candidate separating set S that

will serve as a certificate to an affirmative answer to our problem instance (G, k, t, v_f) must be composed of the union of a set of these minimal separators.

There is a unique path from each leaf to the closest vertex of $\mathcal{T}(v_f)$. Let v_i, v_j be vertices on that path in the tree, and let S_i, S_j be their corresponding minimal vertex separators in G. Without loss of generality, let v_i be closer to $\mathcal{T}(v_f)$ than v_j is. Then the set of vertices separated from v_f in $G - S_i$ is at least as large as the set of vertices separated from v_f in $G - (S_i \cup S_j)$. Thus in a candidate separating set S we need include only at most one minimal separator corresponding to a vertex on the unique path from each leaf to the subtree $\mathcal{T}(v_f)$. There are at most ℓ such paths, so there are $\binom{n^2}{\ell} = \mathcal{O}(n^{2\ell})$ possible combinations of minimal separators to consider when constructing candidate solutions to our problem instance (G, k, t, v_f).

Given a particular candidate separator S, we can check if it provides a certificate for an affirmative answer to (G, k, t, v_f) by checking to see if $|S| \leqslant k$, and if the number of vertices separated from v_f in $G - S$ is at least t in $\mathcal{O}(n)$ time.

Thus we can generate all candidate solutions in time $\mathcal{O}(n^{2\ell})$, and check the feasibility of each in $\mathcal{O}(n)$, giving an overall running time of $\mathcal{O}(n^{2\ell+1})$. □

As a special case for leafage $\ell = 2$ this argument gives us an algorithm to solve the FIREBREAK problem in interval graphs in time $\mathcal{O}(n^5)$. We can do somewhat better in this case using a representation-construction algorithm due to Booth and Lueker [11], who give an $\mathcal{O}(|V| + |E|)$ algorithm, which, using the reasoning above, we can use to give an $\mathcal{O}((|V| + |E|)^4)$ algorithm.

4.2   Permutation Graphs

There are a large variety of types of intersection graphs (in fact, every graph is an intersection graph of some set of objects). While Theorem 7 applies to intersection graphs of paths in a tree (which we note includes interval graphs), permutation graphs are not a subclass of this class of intersection graphs. By using a sweeping-for-separators approach we are able to show that the FIREBREAK problem is tractable on permutation graphs as well. We make use of one of the many characterisations of a permutation graph, as in [36]: a *permutation graph* is the intersection graph of straight line segments between two parallel lines.

We give an example of a permutation graph and a corresponding representation in Figure 2. We will make use of several results on these representations to address the FIREBREAK problem on permutation graphs.

Theorem 8. *An instance* (G, k, t, v_f) *of the* FIREBREAK *problem where* G *is a permutation graph on* n *vertices can be solved in time* $\mathcal{O}(n^3 k^2)$.

Proof. As noted by Lemma 1, the case in which $|N(v_f)| \leqslant k$ can be solved in $\mathcal{O}(n)$ time, so we will now consider the case where $|N(v_f)| > k$. A permutation graph has a *representation* in the form of line segments between two parallel horizontal lines, which itself can be created in $\mathcal{O}(n^2)$ time, and in which we can assume without loss of generality that there are no vertical line segments [36].

This representation gives a partition of vertices: those corresponding to line segments to the left of the line segment representing v_f, those corresponding to line segments to the right
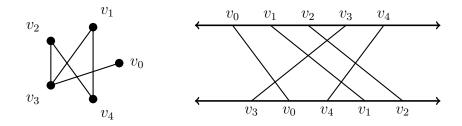
Figure 2: An example of a permutation graph and an associated representation.

of the line segment representing v_f, and those adjacent to v_f (*i.e.*, the lines that intersect the line representing v_f). In this representation there are n line-segment endpoints on the top horizontal line, and n on the bottom (there is one of each for each vertex). If we segment the horizontal lines into portions *between* line segment endpoints, there are therefore $n-1$ segments of each. As described in [10], we can find all minimal separators in a permutation graph from the line segment representation by considering a series of cut-lines, where a *cut-line* is a line between the two horizontal walls with one endpoint on the top horizontal line and one on the bottom, and we consider one cut-line for each pair of top segment and bottom segment. There are thus $\mathcal{O}(n^2)$ such cut-lines, and again due to [10] we know that every minimal separator in the permutation graph consists of the vertices corresponding to the line segments crossed by one of these cut-lines in the representation.

Of these $\mathcal{O}(n^2)$ minimal separators, there are $\mathcal{O}(nk)$ that are of interest to us, namely those that contain at most $(k-1)$ vertices; for convenience we call minimal vertex separators that meet this size constraint $(k-1)$-*small*. Given any $(k-1)$-small minimal separator S defined by the cut-line s, let S_{left} be the subset of vertices to the left of s and let S_{right} be the subset of vertices to the right of s.

Consider the following algorithm that searches for firebreaks (*i.e.*, k-subsets of $V(G)$ that separate v_f from some other vertices) of the form $S \cup T$.

Algorithm 2.

1. Find all minimal separators S with $v_f \in S_{\text{left}}$ (resp. S_{right}) which have size at most k, and denote this set as $\mathcal{L}$ (resp. $\mathcal{R}$).

2. Exhaustively search for all pairs (S,T) such that $S \in \mathcal{L}$, $T \in \mathcal{R}$, $|S| + |T| \leqslant k$ and $|S_{\text{right}}| + |T_{\text{left}}| \geqslant t$.

   If such a pair (S,T) is disjoint and $|S| + |T| = k$, then $S \cup T$ is a firebreak.

   Otherwise, consider the component C in $G - (S \cup T)$ containing v_f. If $|V(C)| - 1 + |S_{\text{right}}| + |T_{\text{left}}| - t + |S \cup T| \geqslant k$ then there is a firebreak as $|V(C)| - 1$ vertices can be removed from this component and $|S_{\text{right}}| + |T_{\text{left}}| - t$ vertices can be removed from other components to produce a firebreak of size k by adding them to $S \cup T$.

3. If no firebreak was found during the exhaustive search, then no firebreak exists.

Suppose some firebreak exists but this algorithm found none. There exist t vertices that can be separated from v_f, possibly some to the left, say L, and some to the right, say R, of v_f. Note that at most one of L and R can be empty. Since L and R are separated from v_f, there must be cut-lines between them and v_f that define minimal separators and hence the algorithm must have found a firebreak.

Since there are $\mathcal{O}(nk)$ minimal separators that are $(k-1)$-small, we can search through the pairs in $\mathcal{O}(n^2k^2)$ time. For each pair (S, T) we may have to find $G - (S \cup T)$ and count $|V(C)|$, $|S_{\mathsf{right}}|$, and $|T_{\mathsf{left}}|$. This adds a factor of n and thus the problem can be solved in $\mathcal{O}(n^3k^2)$ time. $\qquad\square$

5   Acknowledgements

References

[1] B. Addis, M. Di Summa and A. Grosso. Identifying critical nodes in undirected graphs: Complexity results and polynomial algorithms for the case of bounded treewidth. *Discrete Applied Math.* 161 (2013) 2349–2360.

[2] S. Arnborg. Efficient algorithms for combinatorial problems on graphs with bounded decomposability—a survey. *BIT* 25 (1985) 2–23.

[3] S. Arnborg, J. Lagergren and D. Seese. Easy problems for tree-decomposable graphs. *J. Algorithms* 12 (1991) 308–340.

[4] S. Arnborg and A. Proskurowski. Linear time algorithms for NP-hard problems restricted to partial k-trees. *Discrete Applied Mathematics* 23 (1989) 11–24.

[5] C. Ballester, A. Calvó-Armengol and Y. Zenou. Who's who in networks. Wanted: the key player. *Econometrica* 74 (2006) 1403–1417.

[6] N. Baruah and A.K. Baruah. On a traffic control problem using cut-set of graph. *Int. J. Advanced Networking and Applications* 3 (2012) 1240–1244.

[7] A. Berger, A. Grigoriev and R. van der Zwaan. Complexity and approximability of the k-way vertex cut. *Networks* 63 (2014) 170–178.

[8] H.L. Bodlaender. Dynamic programming on graphs with bounded treewidth. Tech. Rep. RUU-CS-87-22. University of Utrecht (1987).

[9] H.L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.* 25 (1996) 1305–1317.

[10] H.L. Bodlaender, T. Kloks and D. Kratsch. Treewidth and pathwidth of permutation graphs. *SIAM J. Disc. Math.* 8 (1995) 606–616.

[11] K.S. Booth and G.S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *Journal of Computer and System Science* 13 (1976) 335–379.

[12] S.P. Borgatti. The key player problem. In: *Dynamic Social Network Modeling and Analysis: Workshop Summary and Papers* (Breiger, Carley and Pattison (eds.)). National Academies Press, Washington (2003) pp. 241–252.

[13] S.P. Borgatti. Identifying sets of key players in a social network. *Comput. Math. Organiz. Theor.* 12 (2006) 21–34.

[14] R.B. Borie, R.G. Parker and C.A. Tovey. Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families. *Algorithmica* 7 (1992) 555–581.

[15] J. Cheriyan and R. Thurimella. Fast algorithms for k-shredders and k-node connectivity augmentation. *J. Algorithms* 33 (1999) 15–50.

[16] B. Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Information and Computation* 85 (1990) 12–75.

[17] Y. Egawa. k-shredders in k-connected graphs. *J. Graph Theory* 59 (2008) 239–259.

[18] S. Finbow and G. MacGillivray. The firefighter problem: a survey of results, directions and questions. *Australas. J. Combin.* 43 (2009) 57–77.

[19] M.R. Garey and D.S. Johnson. The rectilinear Steiner tree problem is NP-complete. *SIAM J. Appl. Math* 32 (1977) 826–834.

[20] M.C. Golumbic. Algorithmic Graph Theory and Perfect Graphs. *Elsevier Science*, 2004, 2nd Edition.

[21] M. Habib and J. Stacho. Polynomial-Time Algorithm for the Leafage of Chordal Graphs. *Algorithms - ESA 2009, 17th Annual European Symposium*, Copenhagen, Denmark, 2009.

[22] B.L. Hartnell. Firefighter! An application of domination. *25th Manitoba Conference on Combinatorial Mathematics and Combinatorial Computing*, University of Manitoba, Winnipeg, 1995.

[23] A.V. Karzanov and E.A. Timofeev. Efficient algorithm for finding all minimal edge cuts of a nonoriented graph. *Cybernetics* 22 (1986) 156–162.

[24] T. Kloks. Treewidth, Computations and Approximations. *Lecture Notes in Computer Science*, vol. 842. Springer-Verlag, Berlin Heidelberg (1994).

[25] P. Sreenivasa Kumar and C.E. Veni Madhavan Minimal vertex separators of chordal graphs. *Discrete Applied Mathematics* 89 (1998) 155–168.

[26] M. Lalou, M.A. Tahraoui and H. Kheddouci. The critical node detection problem in networks: a survey. *Computer Science Review* 28 (2018) 92–117.

[27] A. Langer, F. Reidl, P. Rossmanith and S. Sikdar. Practical algorithms for MSO model-checking on tree-decomposable graphs. *Computer Science Review* 13–14 (2014) 39–74.

[28] D. Marx. Parameterized graph separation problems. *Theoretical Computer Science* 351 (2006) 394–406.

[29] H. Nagamochi and T. Ibaraki. Algorithmic Aspects of Graph Connectivity. Cambridge University Press, New York (2008).

[30] D. Ortiz-Arroyo and D.M.A. Hussain. An information theory approach to identify sets of key players. In: *Intelligence and Security Informatics* (Ortiz-Arroyo, Larsen, Zeng, Hicks and Wagner (eds.)). Lecture Notes in Computer Science, vol 5376. Springer-Verlag, Berlin (2008) pp. 15–26.

[31] J.S. Provan and M.O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM J. Comput.* 12 (1983) 777–788.

[32] N. Robertson and P.D. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *J. Algorithms* 7 (1986) 309–322.

[33] N. Robertson and P.D. Seymour. Graph minors. X. The disjoint paths problem. *J. Combinatorial Theory – Series B* 63 (1996) 65–110.

[34] M.M. Sathik and A.A. Rasheed. A centrality approach to identify sets of key players in an online weblog. *Int. J. Recent Trends in Engineering* 2 (2009) 85–87.

[35] S. Shen and J.C. Smith. Polynomial-time algorithms for solving a class of critical node problems on trees and series-parallel graphs. *Networks* 60 (2012) 103–119.

[36] J. Spinrad. On comparability and permutation graphs. *SIAM J. Comput.* 14 (1985) 658–670.