Throttling processes equivalent to full throttling on trees

Michael S. Ross*

June 22, 2020

Abstract

Consider a discrete-time process on a graph G where a set B of initial vertices are chosen to be colored blue (the remainder being white) and then a time step consists of every currently blue vertex forcing all of its neighbors to become blue; this process stops when every vertex of the graph is blue, and the process is called full forcing. The full throttling number of G is then defined to be the minimum sum of the cardinality of B and the number of time steps needed to complete the forcing process. On trees, the full throttling number is equivalent to the throttling numbers of several other graph processes, such as positive-semidefinite zero forcing, the game of cops and robbers, and the distance domination number (alternately, the k-radius) of a graph. For all of these, it is known that maximum possible throttling number for a tree on n vertices is somewhere between $1.4502\sqrt{n}$ and $\frac{\sqrt{14}}{2}\sqrt{n}$, with the former exhibited by a family of spiders. After introducing some new ideas and methods for working with throttling on trees, this paper determines the exact full throttling number of all balanced spiders (trees with equal-length paths extending from a center vertex), and proves that their full throttling numbers are bounded above by that of paths of the same order n, which are known to have full throttling number $\lceil \sqrt{2n} - \frac{1}{2} \rceil$.

1 Introduction

The study of throttling graph processes and quantities has its origins as a question about zero forcing [8]. Zero forcing is a process used to bound minimum rank/maximum nullity problems from linear algebra and spectral graph theory [1], and arose independently in the control of quantum systems [7, 18]. The zero forcing number has also been shown [20] to be equivalent to fast mixed graph searching. Further, zero forcing appears as a subprocess in the study of power domination [6, 17, 21], which itself is used as a model for the placement of phase measurement units of electrical networks [13, 14]. Purely in terms of graph theory, the zero forcing process involves selecting some initial vertices to color blue, and then that blueness can spread through the graph under specific conditions, in discrete time steps. Thus the most natural problems that arise are to determine the smallest number of initial vertices that will eventually color the entire graph blue (this is the *zero forcing number* of the graph),

---

*Department of Mathematics, Iowa State University, Ames, IA 50011, USA (msross@iastate.edu)

and to determine how much time the process will take with the smallest number of initial vertices necessary (this is the *zero forcing propagation time* of the graph [15]). Zero forcing throttling seeks to strike a balance between these by first noting that a larger initial set of vertices causes the process to terminate sooner, and then finding an optimal solution that minimizes the sum of the number of initial vertices and the propagation time of the zero forcing process.

This concept of balancing initial resources against a related parameter has since been applied to other graph processes, and was first extended to positive semidefinite zero forcing [9] which is a process used bound positive semidefinite minimum rank/maximum nullity problems. Subsequently, throttling was applied to the game of cops and robbers [5], which has applications to the coordination of mobile autonomous agents [16], routing reconfiguration in networks [10], and graph decompositions [4].

We now present a simplified forcing process called *full forcing*, and the related throttling process, which will be called *full throttling*. Let G be a simple graph, $B \subseteq V(G)$ be the current set of blue vertices, and W the current set of white vertices. Then, in a given time step, the *full forcing color change rule* colors $w \in W$ blue whenever w is the neighbor of some $v \in B$, in which case we say that v *forces* w and write $v \to w$. If multiple vertices are capable of forcing w, a choice of forcing vertex is made. Ultimately, this does not affect the contents of the sets $B^{(k)}$ defined below. The full forcing process begins with an initial set of blue vertices, $B^{(0)} = B \subseteq V(G)$, with all other vertices being white. The full forcing color change rule is applied iteratively, and the set $B^{(k)}$ is defined to be the set of all the vertices that $\bigcup_{i=0}^{k-1} B^{(i)}$ can force independently; the collection of forces that color the vertices in $B^{(k)}$ blue are said to occur during the kth *time-step*. If this process eventually colors all of $V(G)$ blue, the set $B^{(0)}$ is said to be a *full forcing set of G*, and the least k such that $\bigcup_{i=0}^{k} B^{(i)} = V(G)$ is the *full forcing propagation time of B*, denoted by $\mathrm{pt}_f(G; B)$. If $B \subset V(G)$ is not a full forcing forcing set, then $\mathrm{pt}_f(G; S) = \infty$. The *full throttling number of B* is $\mathrm{th}_f(G; B) = |B| + \mathrm{pt}_f(G; B)$, and the *full throttling number G* is

$$\mathrm{th}_f(G) = \min_{B \subseteq V(G)} \mathrm{th}_f(G; B).$$

In the event that $\mathrm{th}_f(G; B) = \mathrm{th}_f(G)$, B is said to be *optimal*, or more specifically an *optimal full throttling set* of G. Note that in a connected graph, every nonempty set of vertices is a full forcing set; and thus the interesting questions about full forcing relate to the propagation time and full throttling numbers.

The *distance between vertices* u, v, noted $\mathrm{dist}(u, v)$ is the length of the shortest path between u and v. Given a set $S \subseteq V(G)$ and a vertex v, the *distance from v to S* is $\mathrm{dist}(v, S) = \min_{u \in S} \mathrm{dist}(u, v)$. The *eccentricity of a set of vertices* $S \subseteq V(G)$ is $\mathrm{ecc}(S) = \max_{v \in V(G)} \mathrm{dist}(v, S)$.

The *distance domination number of G*, given a distance d, is the size of the smallest set B of vertices with $\mathrm{ecc}(B) = d$; this is denoted by $\gamma_d(G)$. With a shift in perspective, the *k-radius* of a graph is $\mathrm{rad}_k(G) = \min_{S \subseteq V, |S|=k} \mathrm{ecc}(S)$. In their respective notations, the throttling numbers of distance domination and k-radius are given by $\min_{d \geq 0} \gamma_d(G) + d$ and $\min_{k \geq 1} k + \mathrm{rad}_k(G)$. The throttling of full forcing, distance domination, and k-radius all minimize the sum of a number of vertices and the eccentricity of that set of vertices, and so throttling for each of these parameters is equivalent.

2

The throttling numbers and analogous definitions have been stated for many other processes, a handful of which are given below. Note that for variants of zero forcing, the sets $B^{(i)}$ are defined as for full forcing above; using the associated forcing rules to determine which vertices are forced.

- For (standard) zero forcing, with $v \in B$ (where B is the current set of blue vertices), and $w \in W = V(G) \setminus B$, $v \to w$ whenever w is the *only* white neighbor of v. Then, the propagation time and throttling number of B on G are $\mathrm{pt}(G; B)$ and $\mathrm{th}(G; B) = |B| + \mathrm{pt}(G; B)$ respectively, and the *throttling number of G* [8] is $\mathrm{th}(G) = \min_{B \subseteq V(G)} \mathrm{th}(G; B)$.

- For positive semidefinite zero forcing, we consider the components $W_1, \ldots, W_k$ of the induced subgraph $G[V \setminus B]$; then $v \in B$ forces $w \in W_i$ whenever w is the only white neighbor of v in $G[B \cup W_i]$. The PSD propagation time and PSD-throttling number of B are $\mathrm{pt}_+(G; B)$ and $\mathrm{th}_+(G; B) = |B| + \mathrm{pt}_+(G; B)$. The *PSD-throttling number of G* [9] is then $\mathrm{th}_+(G) = \min_{B \subseteq V(G)} \mathrm{th}_+(G; B)$.

- For the game of cops and robbers, you are given k cops to place on the graph, and then a robber is placed somewhere in the graph. In a round, you move any number of cops to adjacent vertices –winning the game if a cop occupies the same vertex as the robber– and then the robber can move to an adjacent vertex, trying to evade capture for long as possible. Assuming the robber is always placed optimally and both players move optimally, the k-capture time of G is $\mathrm{capt}_k(G)$, the minimum capture time across all choices of k cops. Then, the *cop throttling number of G* [5] is $\mathrm{th}_c(G) = k + \mathrm{capt}_k(G)$.

A throttling process is said to *have full throttling* whenever it is known to be equivalent to full throttling. As noted above, distance domination and k-radius throttling always have full throttling. Positive semidefinite zero forcing has full throttling on trees, as every white neighbor of a blue vertex must exist in its own unique component, so every blue vertex forces all of its neighbors each round, and thus is identical to full forcing. In [5] it was shown that the cop throttling is equivalent to full throttling on chordal graphs.

Thus, on trees,
$$\mathrm{th}_f(G) = \mathrm{th}_c(G) = \mathrm{th}_+(G).$$

Note that these equalities are not true in general, as it is shown in [5, 9] that across all graphs,
$$\mathrm{th}_f(G) \leq \mathrm{th}_c(G) \leq \mathrm{th}_+(G) \leq \mathrm{th}(G),$$

with specific examples of graphs where the adjacent throttlings differ.

Except where specifically noted, the rest of this paper will discuss full throttling and those processes which have full throttling on trees, so the notation $\mathrm{th}_f(G)$ will be used. It was shown in [9] that full throttling is subtree monotonic on trees, and that for paths of order n, $\mathrm{th}_f(P_n) = \lceil \sqrt{2n} - \frac{1}{2} \rceil$. It was also shown in [5] that if T is a tree with the highest full throttling number among all trees of order n, then $\mathrm{th}_f(P_n) \leq \mathrm{th}_f(T) \leq 2\sqrt{n}$, with only a couple specific trees known to have $\mathrm{th}_f(T) = \mathrm{th}_f(P_n) + 1$, and no known examples of trees with a higher full throttling number. It has since been shown in [12] that there is a family of trees T of order n with $1.45\sqrt{n} \leq \mathrm{th}_f(T)$. These trees are all examples of *spiders*, which are

3

trees that have exactly one vertex with degree higher than 2. Spiders are usually described in terms of lengths of their legs; e.g. $S(7, 6, 2)$ is a tree on 16 vertices, with one *center* vertex adjacent to three disjoint paths of orders 7, 6, and 2. A *balanced spider* is one in which every leg has the same length, and is generally noted by $T_{\alpha,\beta} = S(\beta, \beta, \ldots, \beta)$, where the spider has α legs, each of length β. Consequently, this paper defines a *super-spider* as any spider which has a full throttling number higher than that of the same-order path. The upper bound was also improved in [12], where it was noted that $\mathrm{th}_f(T) \leq \sqrt{14n}/2$ for any tree T of order n, and that for all spiders S of order n, $\mathrm{th}_f(S) \leq \sqrt{3n}$.

In Section 2, we show that full throttling is monotonic for connected minors on trees, define a framework to extend any throttling process to apply to weighted graphs, and present a method for computing the full throttling number of highly symmetric graphs, called *concentration*. In Section 3 we show that $\mathrm{th}_f(T_{\alpha,\beta}) = 1 + \alpha \left\lfloor \sqrt{\frac{2\beta+\alpha+1}{4\alpha}} \right\rfloor + \left\lceil \frac{\beta-\hat{s}}{2\hat{s}+1} \right\rceil$, and prove that there are no balanced super-spiders.

2   New Tools for Full Throttling on Trees

In this section we present a useful fact about the full throttling numbers of paths, strengthen the monotonicity results of [9], and introduce a generalization of throttling processes on (vertex) weighted graphs; which is then used in a new technique for computing the full throttling number of highly symmetric trees, by first reducing them to smaller weighted trees.

Lemma 2.1. *Let $t \in \mathbb{Z}^+$. Then, the longest path with throttling number t is P_{n_t}, where n_t is the t-th triangle number. Consequently, $\mathrm{th}_f(P_n) = \left\lceil \sqrt{2n + \frac{1}{4}} - \frac{1}{2} \right\rceil$.*

Proof. Let n_t be the largest integer for which $\mathrm{th}_f(P_{n_t}) = \left\lceil \sqrt{2n_t} - \frac{1}{2} \right\rceil = t$. Then, n_t is the largest integer such that $\sqrt{2n_t} \leq t + \frac{1}{2}$. By squaring both sides and solving the resulting quadratic, one can see that $n_t = \frac{t(t+1)}{2}$. Thus, n_t is the t-th triangle number, and consequently the throttling number of P_n is the ceiling of the inverse triangle number of n . $\square$

On its own, Lemma 2.1 may appear to be no more than a curiosity. However, this variant of $\mathrm{th}_f(P_n)$ can lead to some elegant simplification when used alongside other throttling formulae, as in the proof of Lemma 3.7. Further, the triangle numbers will appear once again when we use them to construct a family of super-spiders in Proposition 3.5.

It was established in [9] that full throttling (there called PSD-throttling) is subtree monotonic. We extend full throttling monotonicity to all connected minors.

Observation 2.2. *Any connected minor of a tree T can be created using only edge contractions.*

Theorem 2.3. *Let T be a tree, and T' be a connected minor of T. Then,*

$$\mathrm{th}_f(T') \leq \mathrm{th}_f(T).$$

That is, full throttling is connected minor monotonic for trees.

Proof. We need consider only edge contraction by Observation 2.2. Let $uv \in E(T)$, $B \subseteq V(T)$, and B' be the image of B under the edge contraction T/uv. That is, B' contains $B \setminus \{u, v\}$, and contains the new vertex if and only if at least one of u or v are in B. Thus, $|B'| \leq |B|$. Next, consider the propagation of B through T. If that process forces through edge uv, then all subsequent forces in that component will occur one time-step sooner in the propagation of B' through T/uv. If the process does not force through edge uv, $\mathrm{pt}_f(T/uv; B') \leq \mathrm{pt}_f(T; B)$. Thus, for all $uv \in E(T)$, $\mathrm{th}_f(T/uv) \leq \mathrm{th}_f(T)$. $\qquad\square$

We now provide a natural extension of the throttling of *any* forcing process to allow for (vertex) weighted graphs. In fact, this provides a blueprint for any process that involves selecting an initial set of vertices from the weighted graph. We'll refer to the generic processes as X-forcing and X-throttling, and will use pt_X and th_X appropriately.

Definition 2.4. Let (G, w) be a weighted graph, where $w : V(G) \to \mathbb{R}^+$ is a weight function on the vertices of G, and let $B \subseteq V(G)$ be an X-forcing set of G. Define $w(B) = \sum_{v \in B} w(v)$. Then,

$$\mathrm{th}_X(G, w; B) = w(B) + \mathrm{pt}_X(G; B)$$

and the X-*throttling number* of (G, w) is

$$\mathrm{th}_X(G, w) = \min_{B \subseteq V(G)} \mathrm{th}_X(G, w; B).$$

In the event that $\mathrm{th}_X(G, w; B) = \mathrm{th}_X(G, w)$, B is said to be an *optimal* (X-throttling) set for (G, w).

Note that this method of throttling on weighted graphs is also a generalization of *weighted X-throttling* on unweighted graphs, defined for zero forcing and positive semidefinite zero forcing in [8, 9] as

$$\mathrm{th}_f{}^\omega(G) = \min_{B \subseteq V(G)} \left(\omega|B| + \mathrm{pt}_f(G; B) \right),$$

wherein the "weighting" takes the form of a scalar ω multiplied by the size of B, rather than weights on individual vertices.

Observation 2.5. *When the weight function w is constant, i.e., $w(v) = \omega$ for all $v \in V(G)$, X-throttling of the weighted graph (G, w) is equal to ω-weighted X-throttling of the unweighted graph G: $\mathrm{th}_X(G, w) = \mathrm{th}_X{}^\omega(G)$. When the weight function is identically one, the result is ordinary X-throttling.*

Next, we define a method by which we can use full throttling on weighted trees to simplify throttling on unweighted graphs, whenever the process being throttled is equivalent on trees to full throttling.

Definition 2.6. Let (T, w) be a weighted tree and let $v \in V(T)$. Then, the components of $T - v$ are called *branches of T at v*. Branches T_1, T_2 of T at v are called *weight isomorphic* when

1. there is an automorphism σ of T such that for all $x \in V(T) \setminus (V(T_1) \cup V(T_2))$, $\sigma(x) = x$, $\sigma(V(T_1)) = V(T_2)$, and $\sigma(V(T_2)) = V(T_1)$, and

2. for all $x \in V(T_1)$, $w(x) = w(\sigma(x))$.

In this case σ is called a *weight isomorphism*. Given a set $T_v = \{T_1, \ldots, T_\ell\}$ of pairwise weight isomorphic branches of T at $v \in V(T)$ with weight isomorphisms σ_i between $V(T_i)$ and $V(T_1)$ for $i = 2, \ldots, \ell$, a set of vertices B is *weight isomorphic with respect to T_v* whenever $B \cap V(T_i) = \sigma_i(B \cap V(T_1))$ for $i = 2, \ldots, \ell$.

Theorem 2.7. *Let (T, w) be an integer weighted tree. Suppose that $T_v = \{T_1, \ldots, T_\ell\}$ is a set of pairwise weight isomorphic branches of T at $v \in V(T)$ with weight isomorphisms σ_i between $V(T_i)$ and $V(T_1)$ for $i = 2, \ldots, \ell$, and that $w(v) = 1$. Then there is an optimal full forcing set B for (T, w) that is weight isomorphic with respect to T_v.*

Proof. Let B be an optimal full forcing set of (T, w), and define $B_i = B \cap V(T_i)$ for $i = 1, \ldots, \ell$. There are two cases, depending on the role of v.

First, suppose $v \in B$ or v can be forced by a vertex not in $\cup_{i=1}^{\ell} V(T_i)$ in at most $\mathrm{pt}_f(T; B)$ time-steps. In particular, this means the full forcing process happens independently in each T_i. Without loss of generality, $w(B_1) \leq w(B_i)$ for $i = 2, \ldots, \ell$. Since forcing in all branches concludes in at most $\mathrm{pt}_f(T; B)$ time-steps, if $w(B_1) < w(B_i)$ we could replace B_i by $\sigma_i(B_1)$ and B was not optimal. Thus, $w(B_i) = w(B_1)$ for $i = 2, \ldots, \ell$, and we can replace B_i by $\sigma_i(B_1)$ to get an optimal full forcing set that *is* weight isomorphic with respect to T_v.

Next we consider the case where v is forced by a vertex in some T_k, where $1 \leq k \leq \ell$. Let $H_i = T[V(T_i) \cup \{v\}]$ for $i = 1, \ldots, \ell$. We may assume, without loss of generality, that $\{B_i \mid i \neq k\}$ is weight isomorphic with respect to $\{T_i \mid i \neq k\}$. Thus, for the remainder of this proof, assume $i \in \{1, \ldots, \ell\}$ with $i \neq k$. If $w(B_k) \leq w(B_i)$, we could replace B_i by $\sigma_i(\sigma_k^{-1}(B_k))$ -contradicting the optimality of B- so we assume $w(B_k) > w(B_i)$. We may also assume $\mathrm{pt}_f(H_k; B_k) < \mathrm{pt}_f(H_i; B_i)$, or else replacing B_k with $\sigma_k(\sigma_i^{-1}(B_i))$ also contradicts the optimality of B. Now, consider $B' = (B \setminus B_k) \cup \sigma_k(\sigma_i^{-1}(B_i)) \cup \{v\}$. Clearly, $\mathrm{pt}_f(H_i; B_i \cup \{v\}) \leq \mathrm{pt}_f(T; B)$, as any forcing caused by v under propagation from B now occurs sooner. Further, since $w(B_k) > w(B_i)$ and $w(v) = 1$, $w(B') \leq w(B)$. Thus, B' is an optimal full forcing set for (T, w) which is weight isomorphic with respect to T_v. $\qquad \square$

It should be noted that there are cases without the condition "$w(v) = 1$", which do not have a weight isomorphic optimal set, as demonstrated with the following example.

Example 2.8. Consider the balanced spider $T = T_{3,7} := S(7, 7, 7)$ with center vertex c. Let $A = \{x \in V(T) \mid \mathrm{dist}(c, x) = 1\}$, $B = \{x \in V(T) \mid \mathrm{dist}(c, x) = 5\}$, and suppose T is given weight function

$$w(x) = \begin{cases} 1 & \text{if } \mathrm{dist}(c, x) \in \{1, 5\} \\ 10 & \text{otherwise.} \end{cases}$$

First, note that $\mathrm{th}_f(T; A) = w(A) + \mathrm{pt}_f(T; A) = 3 + 6 = 9$, and thus no starting set containing a weight 10 vertex can be optimal. Thus, the only weight isomorphic starting sets that *could* be optimal are A, B, and $A \cup B$, with $\mathrm{th}_f(T; B) = 8$, and $\mathrm{th}_f(T; A \cup B) = 8$. However, if our starting set is $B \cup \{a\}$ where $a \in A$, we get

$$\mathrm{th}_f(T; B \cup \{a\}) = 4 + 3 = 7.$$

Thus, *no* weight isomorphic set is optimal.

This construction uses a low cost vertex v near the center c to force through to vertices in other branches, thereby reducing the overall propagation time. However, if $w(c) = 1$, then replacing v with c cannot increase the cost, as $w(v)$ is a positive integer. Further, the time need for $v \to c$ is the same as $c \to v$, and the vertices in other branches that v was forcing get forced from c sooner, meaning propagation in all other branches finishes in at most the same amount of time.

Definition 2.9. Let (T, w) be an integer weighted tree. Suppose $\{T_1, \ldots, T_\ell\}$ is a maximal set of pairwise weight isomorphic branches of T at vertex $v \in V(T)$, such that $w(v) = 1$. A *single concentration of T at v* is the weighted tree $T' = T - \{T_2, \ldots, T_\ell\}$ with weight function

$$w'(x) = \begin{cases} \ell w(x) & \text{if } x \in V(T_1) \\ w(x) & \text{otherwise.} \end{cases}$$

Each graph formed by one or more iterations of this process is called a *concentration* of T.

Theorem 2.10. *Let (T, w) be an integer weighted tree, and let (T', w') be a concentration of T. Then,*

$$\text{th}_f(T) = \text{th}_f(T').$$

Proof. Let $T_v = \{T_1, \ldots, T_\ell\}$ be the pairwise weight isomorphic branches of T that are concentrated in T'. Notice that each set vertices in T that is weight isomorphic with respect to T_v corresponds to exactly one set of vertices in T'. By Theorem 2.7, there is an optimal set B for T that is weight isomorphic with respect to T_v. Let B' be the subset of $V(T')$ corresponding to B. Clearly, $w(B) = w'(B')$ and $\text{pt}_f(T; B) = \text{pt}_f(T'; B')$, so $\text{th}_f(T) \geq \text{th}_f(T')$.

Similarly, let B' be an optimal full throttling set of T', and B be the corresponding set of vertices in T, which is weight isomorphic with respect to T_v. Again, $w(B) = w'(B')$ and $\text{pt}_f(T; B) = \text{pt}_f(T'; B')$, so $\text{th}_f(T) \leq \text{th}_f(T')$ $\qquad\square$

The concentration approach can simplify proofs and computations of the full throttling number, especially those with a high degree of symmetry.

Example 2.11. Consider T, a full binary tree of height h. Suppose $w(v) = 1$ for all $v \in V(T)$, and let c denote the root vertex of T. Then, consider each vertex at distance $h - 1$ from c. Each has a weight of 1, and has two weight isomorphic branches (just leaves). Performing a concentration then merges each leaf pair, doubling the cost of the vertices in each branch. Then one can move to the vertices at distance $h - 2$ from c, each of which has weight 1, and two weight isomorphic branches which are paths. Again, concentrating these paths doubles the weights of the merged vertices. Iterating this concentration process towards c thus results in a path of length h, with a weight sequence $2^0, 2^1, 2^2, \ldots, 2^h$. Thus by Theorem 2.10 it's easy to see that $\text{th}_f(T') = h + 1$ by choosing the vertex that costs only 1, and thus $\text{th}_f(T) = h + 1$ by choosing the center vertex.

3 Spiders

In [5], Breen et al. give an algorithm that constructs, for any tree T, an initial coloring set $B \subseteq V(T)$ such that $\text{th}_f(T; B) \leq 2\sqrt{n}$. Since [9] noted that all paths have a full throttling number approximately $\sqrt{2}\sqrt{n}$, the authors of [5] posed an interesting question: What is the smallest coefficient μ such that for all trees T, asymptotically $\text{th}_f(T) \lesssim \mu\sqrt{n}$? Or, which trees have the highest full throttling number across all trees on n vertices, and what is that number? It was originally thought by some that balanced spiders might provide a family of examples for which $\text{th}_f(T) \approx \mu\sqrt{n}$ with $\mu > \sqrt{2}$. However, we show in this section that this is not possible, after determining the exact value of the full throttling number of a balanced spider.

Recall that the (unweighted) balanced spider with α legs of order β is $T_{\alpha,\beta}$; which has $\alpha\beta + 1$ vertices. Note that $T_{\alpha,\beta}$ has α weight isomorphic branches at the center vertex c, all of which are paths of order β. Thus, $T_{\alpha,\beta}$ can be concentrated to a weighted path of order $\beta+1$, wherein one end vertex (which inherits the label c) has weight one, and all other vertices have weight α.

Observation 3.1. *If s vertices in each leg of $T_{\alpha,\beta}$ (i.e. s non-c vertices from the complete concentration) are optimally chosen and c is not chosen, the full forcing propagation time is $\left\lceil \frac{\beta+1-s}{2s} \right\rceil$. If c is chosen, the full forcing propagation time is $\left\lceil \frac{\beta-s}{2s+1} \right\rceil$.*

Lemma 3.2. *Every (unweighted) balanced spider with at least three legs has an optimal full throttling set containing the center vertex.*

Proof. For $\alpha \geq 3$, $\beta, s \geq 1$, define

$$g(\alpha, \beta, s) = \alpha s + \frac{\beta + 1 - s}{2s},$$

which corresponds to the full throttling number when s vertices from each leg of $T_{\alpha,\beta}$ are chosen and c is not. Similarly, for $\alpha \geq 3, \beta \geq 1, s \geq 0$, define

$$h(\alpha, \beta, s) = 1 + \alpha s + \frac{\beta - s}{2s + 1},$$

corresponding to the full throttling number when c is chosen in addition to the s vertices chosen from each leg. It suffices to show that for every triple (α, β, s) with $\alpha \geq 3, \beta, s \geq 1$,

$$h(\alpha, \beta, s) \leq g(\alpha, \beta, s) \quad \text{or} \quad h(\alpha, \beta, s - 1) \leq g(\alpha, \beta, s).$$

Observe that for fixed α and s, both h and g are linear functions in β. The slopes are

$$\left.\frac{dg(\alpha, \beta, s)}{d\beta}\right|_s = \frac{1}{2s}$$

$$\left.\frac{dh(\alpha, \beta, s)}{d\beta}\right|_s = \frac{1}{2s + 1} < \frac{1}{2s}$$

$$\left.\frac{dh(\alpha, \beta, s)}{d\beta}\right|_{s-1} = \frac{1}{2s - 1} > \frac{1}{2s},$$

8

and the intercepts are

$$
\begin{aligned}
g(\alpha, 0, s) &= \alpha s + \frac{1-s}{2s} = \alpha s - \frac{1}{2} + \frac{1}{2s} \\
h(\alpha, 0, s) &= 1 + \alpha s - \frac{s}{2s+1} = 1 + \alpha s - \frac{1}{2} + \frac{1}{4s+2} > \alpha s - \frac{1}{2} + \frac{1}{2s} \\
h(\alpha, 0, s-1) &= 1 - \alpha + \alpha s - \frac{s-1}{2s-1} = 1 - \alpha + \alpha s - \frac{1}{2} - \frac{1}{4s-2} < \alpha s - \frac{1}{2} + \frac{1}{2s}.
\end{aligned}
$$

Fix α and s. Define b_0 to be the value of β for which $h(\alpha, \beta, s) = g(\alpha, \beta, s)$, so $b_0 = -1 + s + 4s^2$. Then, $h(\alpha, \beta, s) \le g(\alpha, \beta, s)$ for $\beta \ge b_0$. Once we show that $h(\alpha, b_0, s-1) \le g(\alpha, b_0, s)$, it follows that $h(\alpha, \beta, s-1) \le g(\alpha, \beta, s)$ for $\beta \le b_0$, completing the proof.

$$
g(\alpha, b_0, s) - h(\alpha, b_0, s-1) = \frac{\alpha(2s-1) - 4s + 1}{2s-1}
$$

Since $2s - 1 \ge 1$ it suffices to show that $1 + \alpha(2s-1) - 4s \ge 0$. Since $\alpha \ge 3$ and $s \ge 1$,

$$
1 + \alpha(2s-1) - 4s \ge 1 + 3(2s-1) - 4s = 1 + 6s - 3 - 4s = 2s - 2 \ge 0. \qquad \square
$$

Theorem 3.3. *For the balanced spider $T = T_{\alpha,\beta}$ with $\alpha \ge 3$, $\mathrm{th}_f(T) = 1 + \alpha \hat{s} + t$ where*

$$
\hat{s} = \left\lfloor \sqrt{\frac{2\beta + \alpha + 1}{4\alpha}} \right\rfloor \qquad and \qquad t = \left\lceil \frac{\beta - \hat{s}}{2\hat{s} + 1} \right\rceil = \left\lceil \frac{\beta + \frac{1}{2}}{2\hat{s} + 1} - \frac{1}{2} \right\rceil.
$$

Proof. By Lemma 3.2, there is an optimal full throttling set B_0 containing the center vertex. Let P_T be the concentration of T at the center. Then the value of t follows from Observation 3.1, and thus $\mathrm{th}_f(P_T; B_0) = 1 + \alpha s + t$ where s is the number of vertices of weight α (that originally came from the legs).

Now, consider the family of real-valued functions

$$
h(\alpha, \beta, s) = 1 + \alpha s + \frac{\beta - s}{2s + 1},
$$

and note that for fixed $\alpha \ge 3$ and s these are linear functions of β.

Observe that the sequence of intercepts $\{h(\alpha, 0, s)\}_{s \in \mathbb{N}}$ is strictly increasing in s, and that the sequence of slopes $\{h'(\alpha, \beta, s)\}_{s \in \mathbb{N}}$ is strictly decreasing, but is always positive. We consider the sequence $\{\beta_s\}_{s=0}^{\infty}$, where β_s is the value for which $h(\alpha, \beta_s, s-1) = h(\alpha, \beta_s, s)$, i.e. the point at which increasing from $s-1$ to s vertices will not raise and may lower the full throttling number, which is also the values of $b \in \mathbb{R}$ at which the linear functions h intersect.

9

$$\begin{aligned}
h(\alpha, \beta_s, s-1) &= h(\alpha, \beta_s, s) \\
1 + \alpha(s-1) + \frac{\beta_s - (s-1)}{2(s-1)+1} &= 1 + \alpha s + \frac{\beta_s - s}{2s+1} \\
\frac{\beta_s - s + 1}{2s - 1} &= \alpha + \frac{\beta_s - s}{2s+1} \\
\frac{\beta_s - s + 1}{2s - 1} &= \frac{2\alpha s + \alpha + \beta_s - s}{2s+1} \\
2\beta_s s - 2s^2 + 2s + \beta_s - s + 1 &= 4\alpha s^2 + 2\alpha s + 2\beta_s s - 2s^2 - 2\alpha s - \alpha - \beta_s + s \\
2\beta_s &= 4\alpha s^2 - \alpha - 1 \\
\beta_s &= 2\alpha s^2 - \frac{\alpha + 1}{2}.
\end{aligned} \tag{1}$$

Solving (1) for s and taking the floor then gives the optimal choice for $\hat{s}$, given any β.

$$\hat{s} = \left\lfloor \sqrt{\frac{2\beta + \alpha + 1}{4\alpha}} \right\rfloor. \qquad \square$$

Here, we define a continuous variant of the full throttling number for balanced spiders, which will be used in the next section. Let

$$t_S(\alpha, \beta) = 1 + \alpha\hat{s} + \hat{t}$$

where $\hat{s}$ is as defined in Theorem 3.3, and $\hat{t} = \frac{\beta + \frac{1}{2}}{2\hat{s}+1} - \frac{1}{2}$. Note that $\hat{t}$ is obtained by removing the ceiling from t in Theorem 3.3.

Corollary 3.4. *The functions $t_S(\alpha, \beta)$ are continuous in β, and $\text{th}_f(T_{\alpha,\beta}) = \lceil t_S(\alpha, \beta) \rceil$.*

Proof. Note that the second statement follows immediately from the fact that α and $\hat{s}$ are integers. For the first, note that for all β, $t_S(\alpha, \beta) = h(\alpha, \beta, \hat{s})$, and so $t_S(\alpha, \beta) = \min_{s \in \mathbb{N}} h(\alpha, \beta, s)$. Finally, recall that when the optimal value of s, changes to $s+1$, it is at the β for which $h(\alpha, \beta, s) = h(\alpha, \beta, s+1)$, and thus $t_S(\alpha, \beta)$ must be continuous. $\square$

In [5], it is shown that the spider $S(4, 3, 2)$ has a higher full throttling number than the path of the same order. Specifically, $\text{th}_f(S(4,3,2)) = 5 = 1 + \text{th}_f(P_{10})$. A computer search of small spiders shows that this is the smallest spider whose full throttling number exceeds that of the path of the same order. This search, which is described in Appendix 1, also produced several thousand spiders that have full throttling numbers one more than that of the path of the same order. For example, the full throttling numbers of next few smallest such spiders $\text{th}_f(S(5,4,3,2)) = \text{th}_f(S(5,4,4,1)) = \text{th}_f(S(6,4,4)) = \text{th}_f(S(7,4,3)) = 6 > 5 = \text{th}_f(P_{15})$ and $\text{th}_f(S(6,5,4,4)) = \text{th}_f(S(7,5,4,3)) = 7 > 6 = \text{th}_f(P_{20})$.

In light of this, we define a *super-spider* as a spider S for which $\text{th}_f(S) > \text{th}_f(P_{|V(S)|})$. We give a simple infinite family of super-spiders (the triangle spiders) with exactly "path plus one" full throttling number below.

Proposition 3.5. *Let* $t \in \mathbb{Z}^+$ *with* $t \geq 4$. *Then, the spider* $S = S(t, t-1, \ldots, 2)$ *with* $t-1$ *legs on* $n = \frac{t(t+1)}{2}$ *vertices has full throttling number* $\text{th}_f(S) = t + 1 = 1 + \text{th}_f(P_n)$.

Proof. Let ℓ_k be the length k leg of S, and let $p = t - i$ be the proposed propagation time of an optimally chosen set B of vertices. First, observe that all legs ℓ_k with $k > p$ *must* contain a blue vertex if propagation is going to conclude on time. Next, note that leg ℓ_p cannot be fully forced by any vertex in ℓ_{p+1}, as the distance from ℓ_{p+1}'s most central vertex and ℓ_p's least central vertex is $p + 1$. Thus, we must either choose a vertex in ℓ_p, or choose the center vertex. As the center vertex will guarantee the forcing of all ℓ_k with $k \leq p$, this is clearly an optimal choice. Thus we have $\text{th}_f(S; B) \geq (i+1) + p = t + 1$. So $\text{th}_f(S) \geq t + 1$. Note that choosing $p = t$ by just coloring the center vertex gives $\text{th}_f(S) \leq t + 1$. Then Lemma 2.1 gives $\text{th}_f(S) = \text{th}_f(P_n) + 1$. $\qquad\square$

Remark 3.6. As an immediate consequence of Proposition 3.5, there is no constant bound on the number of legs a super-spider can have.

Finally, we show that there are no *balanced* super-spiders. To do so, we will examine the continuous analogues of the full throttling number functions for balanced spiders and paths. $t_S(\alpha, \beta)$ is already defined before Corollary 3.4. For paths, recall from Lemma 2.1 that $\text{th}_f(P_{\alpha\beta+1}) = \left\lceil \sqrt{2(\alpha\beta + 1) + \frac{1}{4}} - \frac{1}{2} \right\rceil$. Thus, we define

$$t_P(\alpha, \beta) := \sqrt{2(\alpha\beta + 1) + \frac{1}{4}} - \frac{1}{2}.$$

Since $\text{th}_f(P_{\alpha\beta+1})$ and $\text{th}_f(T_{\alpha,\beta})$ are the respective ceilings of $t_P(\alpha, \beta)$ and $t_S(\alpha, \beta)$, we need only demonstrate that $t_S(\alpha, \beta) \leq t_P(\alpha, \beta)$.

Lemma 3.7. *Let* $\alpha \geq 3$, *and suppose* $t_S(\alpha, \beta) \leq t_P(\alpha, \beta)$ *for some* $\beta \geq \beta_1$, *where* β_1 *is defined in* (1). *Then* $t_S(\alpha, \beta') \leq t_P(\alpha, \beta')$ *for all* $\beta' \geq \beta$.

Proof. Observe that t_S is locally linear in β (except at each β_s), whereas t_P is concave down in β. Thus, we need only show that the inequality holds for the values β_s, where each change in slope occurs. We prove this by examining the average rates of change of the respective functions over the intervals $[\beta_s, \beta_{s+1}]$. As $t_S(\alpha, \beta)$ is linear on each $[\beta_s, \beta_{s+1}]$, we know it has slope $\frac{1}{2s+1}$. Computing the average slope of t_P on each interval is a bit trickier.

$$\frac{t_P(\alpha, \beta_{s+1}) - t_P(\alpha, \beta_s)}{\beta_{s+1} - \beta_s} = \frac{\left(\sqrt{2(\alpha\beta_{s+1} + 1) + \frac{1}{4}} - \frac{1}{2}\right) - \left(\sqrt{2(\alpha\beta_s + 1) + \frac{1}{4}} - \frac{1}{2}\right)}{\beta_{s+1} - \beta_s}$$

$$= \frac{\sqrt{2\alpha\beta_{s+1} + \frac{9}{4}} - \sqrt{2\alpha\beta_s + \frac{9}{4}}}{\beta_{s+1} - \beta_s}$$

$$= \frac{\sqrt{4\alpha^2 s^2 + 8\alpha^2 s + 3\alpha^2 + -\alpha + \frac{9}{4}} - \sqrt{4\alpha^2 s^2 - \alpha^2 - \alpha + \frac{9}{4}}}{4\alpha s + 2\alpha}$$

$$= \frac{\sqrt{16\alpha^2 s^2 + 32\alpha^2 s + 12\alpha^2 - 4\alpha + 9} - \sqrt{16\alpha^2 s^2 - 4\alpha^2 - 4\alpha + 9}}{4\alpha(2s+1)}$$

11

With a little algebraic manipulation, we see that the average rate of change for $t_S(\alpha, \beta)$ is less than the average rate of change of $t_P(\alpha, \beta)$ when the inequality

$$4\alpha \leq \sqrt{16\alpha^2 s^2 + 32\alpha^2 s + 12\alpha^2 - 4\alpha + 9} - \sqrt{16\alpha^2 s^2 - 4\alpha^2 - 4\alpha + 9}$$

holds. To establish this condition, r_{s+1} and r_s will be used as shorthand for the two square roots, respectively. Thus we need to show that

$$(4\alpha + r_s)^2 \leq r_{s+1}^2$$
$$16\alpha^2 + 8\alpha r_s \leq r_{s+1}^2 - r_s^2$$
$$16\alpha^2 + 8\alpha r_s \leq 32\alpha^2 s + 16\alpha^2$$
$$8\alpha r_s \leq 32\alpha^2 s$$
$$r_s^2 \leq 16\alpha^2 s^2$$
$$16\alpha^2 - 4\alpha^2 - 4\alpha + 9 \leq 16\alpha^2 s^2$$
$$9 \leq 4\alpha^2 + 4\alpha$$

Since $\alpha \geq 3$ by hypothesis, the last inequality holds for all balanced spiders. $\qquad \square$

One should note that this inequality is true for all β. However, β_0 is actually negative, and thus has no context within the problem. Hence the initial restriction $\beta \geq \beta_1$.

Lemma 3.8. *For all $\alpha \geq 3$, $t_S(\alpha, \beta_2) \leq t_P(\alpha, \beta_2)$.*

Proof. Note that $\beta_2 = 8\alpha - \frac{\alpha+1}{2}$. Then we need only show that the difference

$$t_P(\alpha, \beta_2) - t_S(\alpha, \beta_2) = \sqrt{2\alpha\beta_2 + \frac{9}{4}} - \left(1 + 2\alpha + \frac{\beta_2 + \frac{1}{2}}{5}\right)$$

$$= \sqrt{2\alpha\left(8\alpha - \frac{\alpha+1}{2}\right) + \frac{9}{4}} - \left(1 + \frac{10\alpha + (8\alpha - \frac{\alpha+1}{2}) + \frac{1}{2}}{5}\right)$$

$$= \sqrt{15\alpha^2 - \alpha + \frac{9}{4}} - \frac{7}{2}\alpha - 1$$

is non-negative, which it is for all $\alpha \geq 1$. $\qquad \square$

Lemma 3.9. *For all $\alpha \geq 5$, $t_S(\alpha, \beta_1) \leq t_P(\alpha, \beta_1)$.*

Proof. Note that $\beta_1 = \frac{3\alpha-1}{2}$. Then, we need only show that the difference

$$t_P(\alpha, \beta_1) - t_S(\alpha, \beta_1) = \sqrt{2\alpha\beta_1 + \frac{9}{4}} - \left(1 + \alpha + \frac{\beta_1 + \frac{1}{2}}{3}\right)$$

$$= \sqrt{2\alpha\left(\frac{3\alpha-1}{2}\right) + \frac{9}{4}} - \left(1 + \alpha + \frac{\left(\frac{3\alpha-1}{2}\right) + \frac{1}{2}}{3}\right)$$

$$= \sqrt{3\alpha^2 - \alpha + \frac{9}{4}} - \frac{3}{2}\alpha - 1$$

is non-negative. which it is for all $\alpha \geq 5$. $\qquad \square$

(a) $\alpha = 3$ (b) $\alpha = 4$

Figure 1: t_S and t_P as functions of β

Theorem 3.10. *There are no balanced super-spiders.*

Proof. Assume first that $\alpha \geq 5$. By Lemma 3.9, $t_S(\alpha, \beta) \leq t_P(\alpha, \beta)$ for all $\beta \geq \beta_1$. Further, since $t_P(\alpha, 0) = t_S(\alpha, 0) = 1$ for all $\alpha \geq 3$, t_P is concave down in β, and t_S is linear in b over the interval $[0, \beta_1]$, we have that $t_S(\alpha, \beta) \leq t_P(\alpha, \beta)$ for all $\beta \geq 0$. Thus, there are no balanced super-spiders on five or more legs, and we need only demonstrate that there are no balanced super-spiders on three or four legs.

Now, suppose $\alpha \in \{3, 4\}$. Then $t_S(\alpha, \beta_1) \geq t_P(\alpha, \beta_1)$, and $t_S(\alpha, \beta_2) \leq t_P(\alpha, \beta_2)$. Thus, for $\alpha \in \{3, 4\}$, there is a point b_1 in the interval $[0, \beta_1]$ where t_S becomes larger than t_P, and there is another point b_2 in the interval $[\beta_1, \beta_2]$ where t_P becomes larger than t_S (See Figure 1). Since Lemma 3.8 proves there are no balanced super spiders with $\beta \geq \beta_2$, any balanced super-spider must have $\alpha \in \{3, 4\}$, and $\beta \in (b_1, b_2)$.

Suppose $\alpha = 3$. Then $\beta_1 = 4$ and $t_S(3, 3) = t_P(3, 3) = 4$, so $b_1 = 3$. On the other end, $\beta_2 = 22$ and $\frac{17}{3} = t_S(3, 6) < t_P(3, 6) = 5.685$, so $b_2 < 6$. Thus, the only integer candidates for β are 4 and 5. Suppose $\alpha = 4$. Then $\beta_1 = 5.5$ and $t_S(4, 5) = t_P(4, 5) = 6$, so $b_1 = 5$. On the other end, $b_2 = 29.5$ and $7 = t_S(4, 7) < t_P(4, 7) \approx 7.132$, so $b_2 < 7$. Thus, the only integer candidate for β is 6.

To summarize, the balanced spiders $T_{3,4}$, $T_{3,5}$, and $T_{4,6}$ are the *only* candidate balanced super-spiders. However, it is easy to verify that each these has a full throttling number *equal* to that of its correlated path, and thus is *not* a super-spider. $\qquad\square$

A   Algorithms, Computations, and Data

The overall process is as follows. Given n, we first compute the full throttling number t of P_n. To iterate through all spiders, we observe that the spiders have a one-to-one correspondence with the partitions of the integer $n - 1$ that have at least three parts. Next, we iterate through all possible values for $|B| = s \leq \text{th}_f(P_n)$. Once the spider and proposed starting size are chosen, the recursive Sage function below determines if the spider can be fully forced within $p = t - s$ time steps.

```
def spidthrot(partlist, cbool, s, p):
    #partlist = Partition representing the spider
    #cbool = Boolean, true if center is already colored
    #s = remaining number of choices for starting set
    #p = proposed propagation time
    plist=list(partlist)          #Convert Partition to list
    plist.sort()                  #Sort legs
    if (not plist) and (cbool or s>0):
                                  #Legs empty and can finish
        return true
    elif(not plist):              #Legs empty but can't finish
        return false
    elif plist and s == 0:        #Legs not empty, can't choose more; s>=0 for rest
        return false
    elif plist[-1] > 2*p +1:      #Longest needs more than 1, can choose more
        l = plist.pop()
        l = l - (2*p+1)
        plist.append(l)
        return spidthrot(plist, cbool, s-1, p)
                                  #Recursive, cover longest, costs 1
    elif plist[-1] == 2*p +1:     #Longest needs full time, w/o center
        plist.pop()
        return spidthrot(plist, cbool, s-1, p)
    elif plist[-1] == 2*p:        #Covering longest includes center
        plist.pop()
        return spidthrot(plist, true, s-1, p)
    elif plist[-1] > p:           #Covering longest cleans up short legs
        l = plist.pop()
        while plist and plist[0]<= 2*p - l:
            plist.pop(0)
        return spidthrot(plist, true, s-1, p)
    else:                         #Have one to spare, and choosing center covers all.
        return true
```

If the spider has full throttling number at most t, we move on to the next spider. If not, we run the recursion for $t \leq t' \leq t + k$ (usually $k = 1$), to determine the spider's full throttling number, with a special message given if the choice of k is too small.

Below is a table containing most of what is known about super-spiders. For all $1 \leq n \leq 74$, n is omitted from the table if there are no super-spiders of order n. The smallest super-spiders for each value of t are given as tuples.

Note that as the full throttling number increases, super-spiders appear sooner (relative to n_t). This suggests a potential way to construct a tree with a full throttling number higher than path plus one. For example, $n_{12} = 78$, but we have a super-spider (with $\text{th}_f(S) = 13$) on 69 vertices, $S(15, 12, 10, 9, 8, 7, 7)$. Thus there are 9 vertices one might cleverly place to get a full throttling number of 14.

n	$\mathrm{th}_f(P_n)$	# of S-Spiders	Examples
10	4	1	$(4,3,2)$
15	5	4	$(5,4,3,2),(5,4,4,1),(6,4,4),(7,4,3)$
20	6	2	$(6,5,4,4),(7,5,4,3)$
21	6	17	-Many-
26	7	3	$(7,6,5,4,3),(9,6,5,5),(10,6,5,4)$
27	7	17	-Many-
28	7	62	-Many-
33	8	2	$(9,7,6,5,5),(10,7,6,5,4)$
34	8	19	-Many-
35	8	77	-Many-
36	8	221	-Many
41	9	5	$(9,8,7,6,5,5),(10,8,7,6,5,4),(12,9,7,6,6)$ $(13,8,7,6,6),(13,9,7,6,5)$
42	9	31	-Many-
43	9	118	-Many-
44	9	330	-Many-
45	9	783	-Many-
49	10	2	$(12,9,8,7,6,6),(13,9,8,7,6,5)$
50	10	14	-Many-
51	10	61	-Many-
52	10	210	-Many-
53	10	595	-Many-
54	10	1399	-Many-
55	10	2920	-Many-
59	11	4	$(12,10,9,8,7,6,6),(13,10,9,8,7,6,5),$ $(15,12,9,8,7,7),(16,12,9,8,7,6)$
60	11	32	-Many-
61	11	131	-Many-
62	11	441	-Many-
63	11	1201	-Many-
64	11	2803	-Many-
65	11	5792	-Many-
66	11	10986	-Many-
69	12	3	$(15,12,10,9,8,7,7),(16,11,10,9,8,7,7),$ $(16,12,10,9,8,7,6)$
70	12	22	-Many-
71	12	104	-Many-
72	12	380	-Many-
73	12	1123	-Many-
74	12	2823	-Many-

References

[1] AIM Minimum Rank – Special Graphs Work Group (F. Barioli, W. Barrett, S. Butler, S. M. Cioabă, D. Cvetković, S. M. Fallat, C. Godsil, W. Haemers, L. Hogben, R. Mikkelson, S. Narayan, O. Pryporova, I. Sciriha, W. So, D. Stevanović, H. van der Holst, K. Vander Meulen, A. Wangsness). Zero forcing sets and the minimum rank of graphs. *Linear Algebra Appl.*, 428 (2008), 1628–1648.

[2] American Institute of Mathematic workshop *Zero Forcing and its Applications.* `http://aimath.org/pastworkshops/zeroforcing.html`

[3] F. Barioli, W. Barrett, S. Fallat, H.T. Hall, L. Hogben, B. Shader, P. van den Driessche, and H. van der Holst. Zero forcing parameters and minimum rank problems. *Linear Algebra Appl.*, 433 (2010), 401–411.

[4] D. Bienstock, P. Seymour, Monotonicity in Graph Searching, J. of Algo. 12 (1991) 239-245.

[5] J. Breen, B. Brimkov, J. Carlson, L. Hogben, K.E. Perry, C. Reinhart. Throttling for the game of cops and robbers on graphs. *Discrete Math.*, 341 (2018) 2418–2430.

[6] D.J. Brueni, L.S. Heath. The PMU placement problem. SIAM J. Discrete Math., 19, 744761, 2005.

[7] D. Burgarth and V. Giovannetti. Full control by locally induced relaxation. *Phys. Rev. Lett.* PRL 99 (2007), 100501.

[8] S. Butler, M. Young. Throttling zero forcing propagation speed on graphs. *Australas. J. Combin.*, 57 (2013), 65–71.

[9] J. Carlson, L. Hogben, J. Kritschgau, K. Lorenzen, M.S. Ross, S. Selken, V. Valle-Martinez. Throttling positive semidefinite zero forcing propagation time on graphs. *Discrete Appl. Math.*, in press, `https://doi.org/10.1016/j.dam.2018.06.017`.

[10] D. Coudert and J.-S. Sereni, "Characterization of graphs and digraphs with small process number," INRIA, Research Report 6285, Sep. 2007. Available: http://hal.inria.fr/inria-00171083/fr/

[11] R. Diestel. *Graph Theory*, 5th edition. Springer, Berlin, 2017.

[12] J. Geneson, Throttling numbers for adversaries on connected graphs. arXiv:1906.07178 (2019)

[13] T.W. Haynes, S.M. Hedetniemi, S.T. Hedetniemi, M.A. Henning. Domination in graphs applied to electric power networks. SIAM J. Discrete Math., 15, 519529, 2002.

[14] T.W. Haynes, S.M. Hedetniemi, P.J. Slater. Fundamentals of domination in graphs applied to electric power networks. Marcel Dekker, New York, 1998.

[15] L. Hogben, M. Huynh, N. Kingsley, S. Meyer, S. Walker, M. Young. Propagation time for zero forcing on a graph, *Discrete Appl. Math.*, 160 (2012), 1994–2005.

[16] D. Ilcinkas, N. Nisse and D. Soguet. The cost of monotonicity in distributed graph searching. Distributed Computing 22 (2009) 117-127

[17] C.S. Liao. Power domination with bounded time constraints. J. Combin. Optim., 31(2), 725742, 2016.

[18] S. Severini. Nondiscriminatory propagation on trees. *J. Physics A*, 41 (2008), 482–002 (Fast Track Communication).

[19] N. Warnberg. Positive semidefinite propagation time. *Discrete Appl. Math.*, 198 (2016) 274–290.

[20] Boting Yang. Fast-mixed searching and related problems on graphs. *Theoret. Comput. Sci.* 507 (2013), 100–113.

[21] M. Zhao, L. Kang, G.J. Chang. Power domination in graphs. Discrete Math. 306, 1812 1816, 2006.