ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230514

Minimum Target Sets in Non-Progressive Threshold Models: When Timing Matters

Hossein Soltani[a], **Ahad N. Zehmakan**[b;*] **and Ataabak B. Hushmandi**[a]

[a]Urmia University of Technology
[b]The Australian National University

Abstract. Let G be a graph, which represents a social network, and suppose each node v has a threshold value $\tau(v)$. Consider an initial configuration, where each node is either positive or negative. In each discrete time step, a node v becomes/remains positive if at least $\tau(v)$ of its neighbors are positive and negative otherwise. A node set $\mathcal{S}$ is a Target Set (TS) whenever the following holds: if $\mathcal{S}$ is fully positive initially, all nodes in the graph become positive eventually. We focus on a generalization of TS, called Timed TS (TTS), where it is permitted to assign a positive state to a node at any step of the process, rather than just at the beginning.

We provide graph structures for which the minimum TTS is significantly smaller than the minimum TS, indicating that timing is an essential aspect of successful target selection strategies. Furthermore, we prove tight bounds on the minimum size of a TTS in terms of the number of nodes and maximum degree when the thresholds are assigned based on the majority rule.

We show that the problem of determining the minimum size of a TTS is NP-hard and provide an Integer Linear Programming formulation and a greedy algorithm. We evaluate the performance of our algorithm by conducting experiments on various synthetic and real-world networks. We also present a linear-time exact algorithm for trees.

1 Introduction

Over the past few decades, the world has experienced an extreme surge in the proliferation of online social networks. These platforms have emerged as an omnipresent facet of contemporary societies, enabling people to forge connections with their peers, aggregate information, and express their opinions. As such, these networks have become the principal conduits for the rapid dissemination of information, facilitating the fluid formation of opinions through online interactions. Consequently, marketing firms and political campaigns frequently exploit social networks to achieve their desired outcomes, cf. [27]. These entities harness the power of these platforms to advertise new consumer goods and promote political factions. The underlying strategy often revolves around the notion that pinpointing a select group of influential individuals within a given community could trigger a massive ripple effect of influence across the network at large. This has catalyzed a growing interest in the quantitative analysis of opinion diffusion and collective decision-making mechanisms, cf. [24, 20, 15].

From a theoretical standpoint, it is pertinent to introduce and investigate mathematical models of influence diffusion, which simulate the process of how individuals revise their opinions and how the influence disseminates through social interactions. The majority of the proposed models utilize a graph, denoted by G, to model the interactions between members of a community, cf. [36]. The graph, intended to represent a social network, features each node as an individual, with an edge representing a relationship between individuals such as friendship, collaboration, or mentorship. Furthermore, each node is typically assigned a binary state, representing a positive or negative stance regarding a specific topic or the status on the adoption of a novel technological product. Thereafter, nodes continue to update their states as a function of their neighboring nodes' states.

One category of models which has gained significant popularity is the class of threshold models, cf. [36, 24, 1]. These models entail that each node v possesses a distinct threshold value $\tau(v)$ and updates its opinion to a positive state only when the number of its positive neighbors exceeds the stipulated threshold. In these models, while peer pressure could potentially influence a node's decision-making process, nodes can exhibit varying degrees of resistance, where those with higher threshold values require a greater number of positive connections to adopt a positive state.

The majority of threshold based models explored in previous research fall under the umbrella of two categories: progressive and non-progressive, cf. [15, 27, 44]. Progressive models are designed to simulate situations where states evolve in a fixed direction, i.e., once a node assumes a positive disposition, it remains positive indefinitely. This type of dynamic is particularly suited to scenarios where nodes transition from an uninformed (negative) to an informed (positive) state, or where a node adopts a new technology (i.e., switches from negative to positive). Conversely, in the non-progressive setting, nodes possess the capability of oscillating between positive and negative states. In this context, a node's state represents its stance on a given topic (such as levying additional taxes on alcoholic beverages), favoring one of two political parties, or embracing one of two competing services.

A set of nodes whose agreement on positive state results in the whole (or a large body of) network eventually adopting a positive stance is called a *target set*, cf. [15]. In order to acquire insights into the most effective manipulation strategies for controlling the outcome of opinion formation dynamics, the problem of identifying the minimum size of a target set has extensively been examined. This problem is commonly referred to as *target set selection* [1] or *influence maximization* [27], depending on its exact formulation, and has

* Corresponding Author. Email: ahadn.zehmakan@anu.edu.au.
** The extended version of this paper can be found in [41].

yielded a plethora of hardness, algorithmic, and combinatorial findings, cf. [36, 24, 20, 15, 27].

In the present work, we introduce a generalization of a target set, where we allow the nodes to be targeted at different steps of the process (rather than all at once). Such a set of targeted nodes is called a *timed target set*. Some prior work has considered the framework where the manipulator intervenes in the process in a more dynamic fashion, but their setup is fundamentally different from ours, in terms of the underlying diffusion process, the permitted intervention operations, and the manipulator's objectives, cf. [16, 3, 29, 45].

We investigate the problem of finding the minimum size of a timed target set in a non-progressive threshold model. We provide some hardness results, propose a greedy algorithm whose performance is evaluated on real and synthetic graph data, present an exact linear-time algorithm for trees, and prove tight bounds on the minimum size of a timed target set in terms of different graph parameters.

Outline. In the rest of this section, we first give some basic definitions in Subsection 1.1. Building on that, we present our contributions and give an overview of related work in the following two subsections. Our bounds on the minimum size of a timed target set are proven in Section 2. Our complexity and algorithmic results are given in Section 3.

1.1 Preliminaries

Consider a simple undirected graph $G = (V, E)$. We let $n := |V|$, $m := |E|$ and use the shorthand vu (or uv) for an edge $\{v, u\} \in E$. Let $N(v) := \{u \in V : vu \in E\}$ be the *neighborhood* of v and $N[v] = N(v) \cup \{v\}$ be the *closed neighborhood* of v. $d(v) := |N(v)|$ denotes the degree of v and Δ stands for the maximum degree in G. Furthermore, for $D \subseteq V$, let $d_D(v) := |N(v) \cap D|$. We say G is an *even* graph if $d(v)$ is even for every node v in G. For $D \subseteq V$, the induced subgraph of G on D is denoted by $G[D]$, and $G \setminus D$ stands for the induced subgraph on $V \setminus D$. In case of $D = \{v\}$ for some $v \in V$, we use the notation $G \setminus v$ instead of $G \setminus \{v\}$.

By the *threshold assignment* for the nodes of a graph G, we mean a function $\tau : V \longrightarrow \mathbb{N} \cup \{0\}$ such that for each node $v \in V$ the inequality $0 \le \tau(v) \le d(v)$ holds. Some special choices of the threshold assignment are the *strict majority* $\tau(v) = \lceil (d(v) + 1)/2 \rceil$ and *simple majority* $\tau(v) = \lceil d(v)/2 \rceil$.

Consider a pair (G, τ) and an initial *configuration* where each node is either *positive* or *negative*. In the *progressive threshold model*, in each discrete time step, a negative node v becomes positive if at least $\tau(v)$ of its neighbors are positive and positive nodes remain unchanged. In the *non-progressive threshold model*, a node v becomes positive if at least $\tau(v)$ of its neighbors are positive and becomes negative otherwise. Note that in the non-progressive model, nodes can switch from positive to negative while this is not possible in the progressive model. Furthermore, we define $\mathcal{A}_i$ to be the set of positive nodes in the i-th step of the process.

A set $\mathcal{S} \subseteq V$ is called a *target set* (TS) whenever the following holds: If $\mathcal{S}$ is fully positive, then all nodes become positive after some steps, i.e., if $\mathcal{A}_0 = \mathcal{S}$, then $\mathcal{A}_i = V$ for some $i \in \mathbb{N}$. For a pair (G, τ), the minimum size of a TS in the progressive and non-progressive model is denoted by $\overrightarrow{MT}(G, \tau)$ and $\overleftrightarrow{MT}(G, \tau)$, respectively. Note that we use a forward arrow for the progressive model and a bidirectional arrow for the non-progressive model.

According to the definition of a TS, a manipulator targets a set of nodes at once. However, it is sensible to consider the set-up where the manipulator can target nodes at different steps of the process. We capture this by introducing the concept of a *timed target set* (TTS), defined below.

Definition 1. *For a pair (G, τ), the finite sequence $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_k$ of subsets of V, for some integer k, is said to be a TTS in the non-progressive model when there is a sequence $Q_0, Q_1, \ldots, Q_k$ of subsets of V such that : (i) $Q_0 = \emptyset$; (ii) $v \in Q_i$ for $i \ge 1$ if and only if $|N(v) \cap (\mathcal{S}_{i-1} \cup Q_{i-1})| \ge \tau(v)$; (iii) $\mathcal{S}_k = \emptyset$; (iv) $Q_k = V$. We denote the minimum size of a TTS in the non-progressive model with $\overleftrightarrow{MTT}(G, \tau)$, where the size of a TTS is equal to $\sum_{i=0}^{k} |\mathcal{S}_i|$.*

Note that $\mathcal{S}_i \cup Q_i$ is the set of positive nodes in the i-th step, i.e., $\mathcal{A}_i$. The nodes of Q_i become positive using their positive neighbors in step $i-1$ (i.e., their neighbors in $\mathcal{S}_{i-1} \cup Q_{i-1}$) but the nodes of $\mathcal{S}_i$ are chosen to be positive in step i by the manipulator.

We do not define TTS for the progressive model since the ability to target nodes at different steps does not give the manipulator any extra power. This is because there is no benefit for a manipulator to target a node in some step i, for $i \ge 1$, instead of targeting it in step 0.

As a warm-up, let us provide the example below.

Example 1. *Consider the complete bipartite graph $K_{1,n-1}$ with partite sets $X = \{x\}$ and $Y = \{y_1, y_2, \ldots, y_{n-1}\}$. Let τ be the strict majority threshold, i.e., $\tau(v) = \lceil (d(v) + 1)/2 \rceil$ for every node v.*

- *$\mathcal{S} = X$ is a minimum size TS in the progressive model and thus $\overrightarrow{MT}(K_{1,n-1}, \tau) = 1$.*
- *Let $\mathcal{S} = X \cup Y'$ where Y' is any subset of Y of cardinality $\tau(x)$. It is easy to see that $\mathcal{S}$ is a minimum size TS in the non-progressive model, which yields $\overleftrightarrow{MT}(K_{1,n-1}, \tau) = \tau(x) + 1 = \lceil \frac{n}{2} \rceil + 1$.*
- *The sequence $\mathcal{S}_0, \mathcal{S}_1, \mathcal{S}_2$ with $\mathcal{S}_0 = \mathcal{S}_1 = \{x\}$ and $\mathcal{S}_2 = \emptyset$ is a TTS of size 2 and there is no TTS of size 1. Hence, $\overleftrightarrow{MTT}(K_{1,n-1}, \tau) = 2$.*

1.2 Our Contribution

We focus on the minimum size of a TTS in the non-progressive threshold model, i.e., $\overleftrightarrow{MTT}(G, \tau)$. We present tight bounds, prove hardness results, and provide approximation and exact algorithms for general and special classes of graphs.

Timing Matters. In the non-progressive model, TTS has two advantages over the original TS: (i) the nodes can be targeted at different steps (ii) a node can be targeted more than once. Example 1 demonstrates that these two advantages amplify the power of a manipulator significantly since $\overleftrightarrow{MTT}(K_{1,n-1}, \tau)$ is much smaller than $\overleftrightarrow{MT}(K_{1,n-1}, \tau)$. What if we require that $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset$ for every two distinct sets $\mathcal{S}_i$ and $\mathcal{S}_j$ in the definition of a TTS, i.e., take away the advantage (ii)? We prove that the advantage (i) suffices to make the manipulator substantially stronger for some classes of graphs and threshold assignments. Thus, targeting nodes at appropriate time is very crucial for successful manipulation and this paper is the first to consider this fundamental aspect of the target set selection.

Tight Bounds on $\overleftrightarrow{MTT}(G, \tau)$ for Strict Majority. We prove that $\overleftrightarrow{MTT}(G, \tau) \ge 2n/(\Delta + 1)$ when τ is the strict majority and this bound is the best possible. (This extends a result from [24].) We first prove this bound for bipartite graphs using some combinatorial and potential function arguments and then extend to the general case. This result implies that for bounded-degree graphs, a naïve algorithm which simply targets all nodes has a constant approximation ratio. When the graph G is even, we provide the stronger bound of $4n/(\Delta+2)$. The improvement might seem negligible at the first glance, but it is actually quite impactful. For example, for a cycle C_n, the first bound is equal to $2n/3$, but the second one gives the tight bound of n. Furthermore, while requiring the graph to be even might seem very demanding, it actually captures some important graph classes such as the d-dimensional torus. Determining $\overleftrightarrow{MT}(G,\tau)$ and $\overrightarrow{MT}(G,\tau)$ for the d-dimensional torus was studied extensively by prior work, due to certain applications in statistical physics, and the exact answer was proven only after a long line of papers, cf. [34].

Inapproximability Result. We prove that the problem of finding $\overleftrightarrow{MTT}(G,\tau)$ for a given pair (G,τ) cannot be approximated within a ratio of $\mathcal{O}(2^{\log^{1-\epsilon} n})$ for any fixed $\epsilon > 0$ unless $NP \subseteq DTIME(n^{polylog(n)})$, by a polynomial time reduction from the progressive variant.

Integer Linear Programming Formulation. A standard approach to tackle an NP-hard problem is to formulate it as an Integer Linear Program (ILP). Then, we can use standard and powerful ILP solvers to solve small-size problems. We provide an ILP formulation for the problem of finding $\overleftrightarrow{MTT}(G,\tau)$.

Greedy Algorithm and Experiments. We propose a greedy algorithm which finds a TTS for a given pair (G,τ) and prove its correctness. Then, we provide the outcome of our experiments on various synthetic and real-world graph data. Our experiments on small synthetic networks demonstrate that the minimum size of a TTS is strictly smaller than the minimum size of a TS in most cases, confirming that the effect of timing is not restricted to tailored graphs (such as the one given in Example 1). To find the optimal solutions, we rely on our ILP formulation. Furthermore, we observe that our greedy approach returns almost optimal solutions in most cases. We also compare the outcome of our greedy algorithm against an analogous greedy algorithm for TS on real-world networks, such as Facebook and Twitter, and observe a 13% improvement.

Exact Linear-Time Algorithm for Trees. It is known, cf. [15, 13, 8], that if we limit ourselves to trees, then the problem of determining the minimum size of a TS in the progressive model (i.e., $\overrightarrow{MT}(G,\tau)$) is no longer NP-hard and an exact polynomial time algorithm exists. It was left as an open problem in [8] whether a similar result could be proven for the non-progressive variants. Recently, it was shown [39] that if $\tau(v) \in \{0,1,d(v)\}$ for every node v, then the problem for $\overleftrightarrow{MT}(G,\tau)$ is tractable on trees. We make progress on this front by providing a linear-time algorithm which outputs $\overleftrightarrow{MTT}(G,\tau)$ when G is a tree and for any choice of τ. Our algorithm can be interpreted as a dynamic programming approach. It is worth to emphasize that algorithms for trees are not only theoretically interesting, but also the pathway to fixed-parameter tractable algorithms in terms of treewidth, cf. [8], which are relevant from a practical perspective too.

Proof Techniques. Whilst for some of our results such as hardness proof and greedy algorithm, we leverage the rich literature on TS, for others we need to develop several novel proof techniques. In particular, we introduce several new combinatorial and graph tools for the proof of our bounds. Furthermore, we devise novel techniques to establish a linear-time algorithm for trees, which in fact might be useful to settle the problem for TS (since it is only resolved for a very constrained setup as mentioned above.)

1.3 Related Work

Numerous models have been developed and studied to gain more insights into the mechanisms and general principles driving the opinion formation and information spreading among the members of a community, cf. [23, 12, 11, 5, 9]. In the plethora of opinion diffusion models, the threshold models have received a substantial amount of attention. While both the progressive and non-progressive threshold models had been studied in the earlier work, cf. [37, 4], they were popularized by the seminal work of Kempe, Kleinberg, and Tardos [27].

Convergence Properties. In the progressive threshold model, it is straightforward to observe that the process reaches a fixed configuration (where no node can update) in at most n steps. For the non-progressive variant, Goles and Olivos [26] proved that the process always reaches a cycle of configurations of length one or two (i.e., a fixed configuration or switching between two configurations). Furthermore, this happens in $\mathcal{O}(m)$ steps, where m is the number of edges, according to [38]. Stronger bounds are known for special cases. For example, a logarithmic upper bound is proven in [42] for graphs with strong expansion properties and the simple majority threshold. The convergence properties have also been studied for directed acyclic graphs, cf. [18], and when a bias towards a superior opinion is present, cf. [31].

Bounds. There is a large body of research whose main goal is to find tight bounds on the minimum size of a TS in threshold models. Some prior work has investigated this for special classes of graphs, such as the d-dimensional torus [34] and random regular graphs [42]. However, the main focus has been devoted to discovering sharp bounds in terms of various graph parameters such as the number of nodes [28, 6], girth [19], maximum/minimum degree [24], expansion [42], vertex-transitivity [35], and the minimum size of a feedback vertex set [2].

Hardness. The problem of determining the minimum size of a TS in the progressive threshold model has been investigated extensively, and it is known to be NP-hard even for some special choices of the threshold assignment and the input graph. Notably, it was proven in [15] that the problem cannot be approximated within the ratio of $\mathcal{O}(2^{\log^{1-\epsilon} n})$, for any fixed constant $\epsilon > 0$, unless $NP \subseteq DTIME(n^{polylog(n)})$, even if we limit ourselves to simple majority threshold assignment and regular graphs. Hardness results are also known for the non-progressive variant. For the simple majority threshold assignment, the problem cannot be approximated within a factor of $\log\Delta \log\log\Delta$, unless $P = NP$, according to [33]. For more hardness results also see [10, 36, 43].

Algorithms. For certain classes of graphs and threshold assignments, the problem of finding the minimum size of a TS becomes tractable. For the progressive variant, it is proven that there is a polynomial time algorithm when the input graph is a tree, cf. [15, 13, 8]. This was generalized to block-cactus graphs in [17]. The problem also is tractable when the feedback edge set number is small, cf. [36]. Following up on an open problem from [8], recently it was shown in [39] that in the non-progressive variant if $\tau(v) \in \{0,1,d(v)\}$ for every node v, then the problem is polynomial time solvable for trees. It is still open whether a polynomial time algorithm for the general threshold assignment on trees exists or not. (As mentioned in Section 1.2, we provide a linear-time algorithm for TTS for any choice of the threshold assignment.) Furthermore, it is known that for the non-progressive model with the simple majority threshold, there exist a $(\log\Delta)$-approximation algorithm for general graphs [33] and an exact linear-time solution for cycle and path graphs [21]. Finally, the integer linear programming formulation of the problem has been studied by prior work, cf. [1, 40, 14].

2 Bounding Minimum Size of a TTS

Let a *disjoint* TTS be the same as a TTS except that a node cannot be targeted more than once (i.e., $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset$ for every two distinct sets $\mathcal{S}_i$ and $\mathcal{S}_j$ in the definition of a TTS) and define $\overleftrightarrow{MDTT}(G,\tau)$ to be the minimum size of a disjoint TTS in the non-progressive model. In Theorem 1, we prove that there are graphs and threshold assignments for which $\overleftrightarrow{MDTT}(G,\tau)$ is asymptotically smaller than $\overleftrightarrow{MT}(G,\tau)$. This indicates that from the two advantages that TTS has over TS (namely, (i) the nodes can be targeted at different steps (ii) a node can be targeted more than once), advantage (i) solely suffices to make the manipulator substantially more powerful.

Theorem 1. *There are arbitrarily large graphs G such that $\overleftrightarrow{MT}(G,\tau) = \omega(\overleftrightarrow{MDTT}(G,\tau))$, where τ is strict majority.*

Proof Sketch. For an arbitrary integer κ, let set L_i, for $1 \le i \le \kappa$, contains i nodes. Then, add an edge between every node in L_i and L_{i+1}, for $1 \le i \le \kappa - 1$. Finally, attach two leaves to the node in L_1 to construct a graph G with $n = \Theta(\kappa^2)$ nodes. (See Figure 1 for an example.)

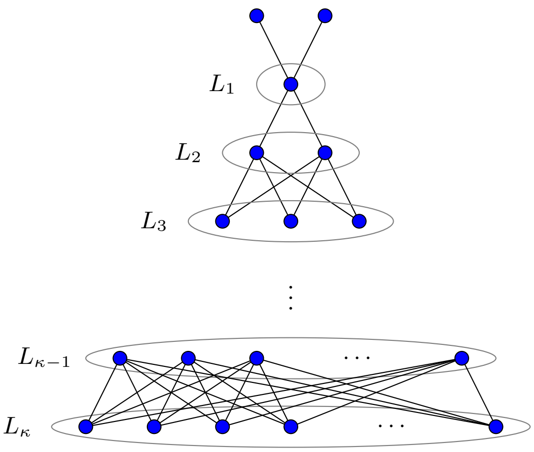

Figure 1: A graph where $\overleftrightarrow{MT}(G,\tau) = \omega(\overleftrightarrow{MDTT}(G,\tau))$.

Suppose that $\mathcal{S}_0 = L_\kappa \cup L'_{\kappa-1}$ in which $L'_{\kappa-1}$ is any subset of $L_{\kappa-1}$ of cardinality $\lceil \frac{\kappa}{2} \rceil$. Also assume that $\mathcal{S}_{\kappa-2}$ consists of the node of L_1 and one of its leaf neighbors. For $1 \le i \ne \kappa - 2 \le \kappa - 1$ set $\mathcal{S}_i = \emptyset$. It is straightforward to check that $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_{\kappa-1}$ is a disjoint TTS of size $\kappa + \lceil \frac{\kappa}{2} \rceil + 2$. This implies that $\overleftrightarrow{MDTT}(G,\tau) = \mathcal{O}(\sqrt{n})$.

Let $\mathcal{S}$ be a TS in the non-progressive model with strict majority. We claim that $|(L_{i-1} \cup L_{i+1}) \cap \mathcal{S}|$ must be at least $i+1$, for $2 \le i \le \kappa - 1$. Thus, we have $|\mathcal{S}| = \Omega(n)$, which implies that $\overleftrightarrow{MT}(G,\tau) = \Omega(n)$. For the sake of contradiction, assume that $|(L_{i-1} \cup L_{i+1}) \cap \mathcal{S}| < i+1$ for some $2 \le i \le \kappa - 1$. Consider the initial configuration where only the nodes in $\mathcal{S}$ are positive. We observe that L_i becomes fully negative after one step. One step after that, L_{i-1} becomes fully negative and so on until the only node in L_1 is negative. Once this happens, in all the following steps either the node in L_1 or its two leaf neighbors will be negative, regardless of the state of other nodes. This is in contradiction with $\mathcal{S}$ being a TS. (A full proof is given in the extended version..) □

2.1 General Graphs

We first prove that $\overleftrightarrow{MTT}(G,\tau) \ge \frac{2n}{\Delta+1}$ for the strict majority threshold assignment if G is bipartite in Theorem 2 (which is based on Lemma 1). Then, we provide Theorem 3 which sets a connection between the value of $\overleftrightarrow{MTT}(G,\tau)$ in bipartite graphs and general graphs. Combining these two theorems gives us our desired bound for general graphs in Theorem 4. The full proofs for these theorems are given in the extended version.

Lemma 1. *Let $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_k$ be a TTS of (G,τ) for a bipartite graph G with partite sets X and Y. Define $\mathcal{S}_e := \mathcal{S}_0 \cup \mathcal{S}_2 \cup \mathcal{S}_4 \cup \cdots$ and $\mathcal{S}_o := \mathcal{S}_1 \cup \mathcal{S}_3 \cup \mathcal{S}_5 \cup \cdots$. If for $D \subseteq V$ we have $D \cap \mathcal{S}_e \cap X = \emptyset$ and $D \cap \mathcal{S}_o \cap Y = \emptyset$ (or $D \cap \mathcal{S}_o \cap X = \emptyset$ and $D \cap \mathcal{S}_e \cap Y = \emptyset$), then $|E(G[D])| \le \sum_{u \in D} (d(u) - \tau(u))$.*

Proof Sketch. Let us first prove Claim 1, which is the main ingredient of this proof.

Claim 1. *If D is nonempty, then there is a node $u \in D$ such that $d_D(u) \le d(u) - \tau(u)$.*

Assume that $D \cap \mathcal{S}_e \cap X = \emptyset$ and $D \cap \mathcal{S}_o \cap Y = \emptyset$ (the proof is analogous for the other case). For the sake of contradiction, assume that $d_D(u) > d(u) - \tau(u)$ for all $u \in D$. One can show that for every odd i, $(D \cap Y) \cap \mathcal{A}_i = \emptyset$ and for every even i, $(D \cap X) \cap \mathcal{A}_i = \emptyset$. Since $D \ne \emptyset$, this contradicts the fact that $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_k$ is a TTS. This finishes the proof of Claim 1.

The statement of the lemma is trivial for $D = \emptyset$. For $|D| \ge 1$ the proof is by induction on $|D|$. The base case of $|D| = 1$ is straightforward. Assume that the inequality holds for $|D| < k$. Using Claim 1, there exists a node $v \in D$ such that $d_D(v) \le d(v) - \tau(v)$. Applying the induction hypothesis for $D' := D \setminus \{v\}$ and some small calculations finish the proof. □

Theorem 2. *$\overleftrightarrow{MTT}(G,\tau) \ge \frac{2n}{\Delta+1}$ if G is bipartite and τ is the strict majority.*

Proof Sketch. Let X and Y be the partite sets of G. Assume that $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_k$ is a TTS of (G,τ). Let $\mathcal{S}_o$ and $\mathcal{S}_e$ be as defined in Lemma 1. Furthermore, define $S := \mathcal{S}_o \cup \mathcal{S}_e$, $S' := \mathcal{S}_o \setminus \mathcal{S}_e$, $S'' := \mathcal{S}_e \setminus \mathcal{S}_o$, $S''' := \mathcal{S}_o \cap \mathcal{S}_e$. Let $\mathcal{S}_X := S \cap X$, $\mathcal{S}'_X := S' \cap X$, $\mathcal{S}''_X := S'' \cap X$, $\mathcal{S}'''_X := S''' \cap X$ (and similarly, define $\mathcal{S}_Y$, $\mathcal{S}'_Y$, $\mathcal{S}''_Y$, and $\mathcal{S}'''_Y$). Let $F_1 := I \cup \mathcal{S}'_X \cup \mathcal{S}''_Y$ and $F_2 := I \cup \mathcal{S}''_X \cup \mathcal{S}'_Y$, where $I := V \setminus S$. Both F_1 and F_2 clearly satisfy the conditions of Lemma 1. Combining the two inequalities obtained from applying Lemma 1 and some calculations, we get:

$$\sum_{u \in I} (2\tau(u) - d(u)) \le \sum_{u \in S} (d(u) - \tau(u)) + \sum_{u \in S'''} \tau(u). \quad (1)$$

Using the fact that for the strict majority threshold assignment, $\tau(u) \ge \frac{d(u)+1}{2}$ and some further calculations, we can conclude that $\frac{2n}{\Delta+1} \le |S' \cup S''| + 2\,|S'''| \le \overleftrightarrow{MTT}(G,\tau)$. □

Theorem 3. *Let G be a graph with node set $V(G) := \{v_1, v_2, \ldots, v_n\}$ and τ be a threshold assignment of its nodes. Construct the bipartite graph H with partite sets $X := \{x_1, x_2, \ldots, x_n\}$ and $Y := \{y_1, y_2, \ldots, y_n\}$ whose edge set is $E(H) := \{x_i y_j | v_i v_j \in E(G)\}$. Consider threshold assignment τ' for H with $\tau'(x_i) = \tau'(y_i) = \tau(v_i)$ for $1 \le i \le n$. Then, $\overleftrightarrow{MTT}(H,\tau') = 2\overleftrightarrow{MTT}(G,\tau)$.*

Combining Theorems 2 and 3 gives us Theorem 4.

Theorem 4. $\overleftrightarrow{MTT}(G,\tau) \geq \frac{2n}{\Delta(G)+1}$ *if τ is the strict majority.*

Tightness. According to Example 1, $\overleftrightarrow{MTT}(K_{1,n-1},\tau) = 2$ when τ is the strict majority. This implies that the bound of $\overleftrightarrow{MTT}(K_{1,n-1},\tau) \geq \frac{2n}{\Delta(K_{1,n-1})+1} = \frac{2n}{(n-1)+1} = 2$ is tight.

2.2 Even Graphs

Theorem 5. $\overleftrightarrow{MTT}(G,\tau) \geq \frac{4n}{\Delta+2}$ *if G is even and τ is the strict majority, and this bound is tight.*

Proof Sketch. It suffices to prove the bound for bipartite graphs. Then, we can apply Theorem 3 to extend to general graphs. Thus, let G be bipartite.

Since $d(u)$ is an even number for any node u, we have $\tau(u) = \left\lceil \frac{d(u)+1}{2} \right\rceil = \frac{d(u)+2}{2}$ which implies $2\tau(u) - d(u) = 2$. Plugging this into Equation (1) in the proof of Theorem 2 yields $2|I| \leq \sum_{u\in S}(d(u)-\tau(u)) + \sum_{u\in S'''}\tau(u)$. Executing some calculations similar to the proof of Theorem 2, we can show that $\frac{4n}{\Delta+2} \leq |S' \cup S''| + 2|S'''| \leq \overleftrightarrow{MTT}(G,\tau)$. For a full proof of the theorem (including its tightness), see the extended version. □

3 Algorithms to Find Minimum TTS

We first prove that the TIMED TARGET SET SELECTION problem is hard to approximate. In Subsections 3.2 and 3.3, we provide an ILP formulation and propose a greedy algorithm for the problem, respectively. Then, we provide our experimental findings for real-world and synthetic graph data in Subsection 3.4. Finally, we present our linear time algorithm for trees in Subsection 3.5.

TIMED TARGET SET SELECTION
Input: A graph G and a threshold assignment τ.
Output: The minimum size of a TTS, i.e., $\overleftrightarrow{MTT}(G,\tau)$.

3.1 Hardness Result

Theorem 6. *The* TIMED TARGET SET SELECTION *problem cannot be approximated within a ratio of $\mathcal{O}(2^{\log^{1-\epsilon} n})$ for any fixed $\epsilon > 0$, unless $NP \subseteq DTIME(n^{polylog(n)})$.*

Proof Sketch. For a given pair (H,τ'), construct (G,τ) as follows. For each node $v \in V(H)$, add $\left\lceil \frac{d(v)}{2} \right\rceil$ copies of the complete graph K_2 and connect both nodes of each copy to v. Set $\tau(v) = \tau'(v)$ if $v \in V(H)$ and $\tau(v) = 1$ otherwise. We claim that $\overrightarrow{MT}(H,\tau') = \overleftrightarrow{MTT}(G,\tau)$.

Assume that there is a polynomial time algorithm for finding $\overleftrightarrow{MTT}(G,\tau)$ with the approximation ratio $C2^{\log^{1-\epsilon}|V(G)|}$ for some constants $C,\epsilon > 0$. Since $|V(G)| = \mathcal{O}(|V(H)|^2)$, the above polynomial time reduction gives us a polynomial time algorithm with approximation ratio $C'2^{\log^{1-\epsilon'}|V(H)|}$, for some constants $\epsilon', C' > 0$, for the problem of finding the minimum size of a TS in the progressive model. However, this is not possible according to the results from [15], unless $NP \subseteq DTIME(n^{polylog(n)})$. (Please refer to the extended version for a full proof.) □

3.2 Integer Linear Programming Formulation

Here, we provide an Integer Linear Program (ILP) formulation for the TIMED TARGET SET SELECTION problem. The binary variables x_{vi} and y_{vi} stand for the state of node v in time step i. We have $x_{vi} = 1$ if and only if $v \in \mathcal{S}_i$ and $y_{vi} = 1$ if and only if $v \in Q_i$. So $x_{vi} + y_{vi} \geq 1$ if and only if v is positive in time step i, i.e., $v \in \mathcal{A}_i$. Let $\mathbb{K} := \{1,\cdots,k\}$ and $\mathbb{K}_0 := \mathbb{K} \cup \{0\}$. Furthermore, let us define the constraints

- $C := (d(v)+1-\tau(v))y_{vi} + \tau(v) - 1 \geq \sum_{u\in N(v)}(x_{u(i-1)} + y_{u(i-1)})$
- $C' := \sum_{u\in N(v)}(x_{u(i-1)} + y_{u(i-1)}) - \tau(v)y_{vi} \geq 0.$

$$\begin{array}{lll} \min & \sum_{i=0}^{k}\sum_{v\in V} x_{vi} & \\ \text{s.t.} & C & \forall v \in V \;\; \forall i \in \mathbb{K} \\ & C' & \forall v \in V \;\; \forall i \in \mathbb{K} \\ & x_{vk} + y_{vk} = 1 & \forall v \in V \\ & x_{vi} + y_{vi} \leq 1 & \forall v \in V, i \in \mathbb{K}_0 \\ & y_{v0} = 0 & \forall v \in V \\ & x_{vi} \in \{0,1\} & \forall v \in V \;\; \forall i \in \mathbb{K}_0 \\ & y_{vi} \in \{0,1\} & \forall v \in V \;\; \forall i \in \mathbb{K}_0 \end{array} \tag{2}$$

If the number of positive neighbors of a node v in time step $i-1$ is greater than or equal to $\tau(v)$, then v becomes positive in time step i. This is expressed as constraint C in the ILP. Note that for $y_{vi} = 0$, the constraint C is equal to $\tau(v) > \sum_{u\in N(v)}(x_{u(i-1)} + y_{u(i-1)})$. Furthermore, if the number of positive neighbors of a node v in time step $i-1$ is strictly less than $\tau(v)$, then v becomes negative in time step i (unless we force it to be positive i.e., $x_{vi} = 1$). This is expressed as the constraint C'. We observe that if $y_{vi} = 1$, then the constraint C' is equal to $\sum_{u\in N(v)}(x_{u(i-1)} + y_{u(i-1)}) \geq \tau(v)$. The other constraints and the objective function are self-explanatory.

The ILP (2) formulates the problem of finding the minimum size of a TTS which reaches the fully positive configuration in k steps. To prove this, we need to show that a solution to the ILP corresponds to a TTS of the same size and vice versa. A formal proof of this is given in the extended version, which builds on the observations from above on the connections between the constraints of the ILP and the updating rules.

It is known [38] that the non-progressive model stabilizes in $\mathcal{O}(n^2)$ steps. Putting this in parallel with the fact that a minimum TTS is of size at most n, we conclude that there is always a minimum TTS which reaches the fully positive configuration in $\mathcal{O}(n^3)$ steps. (Some details are left out.) Thus, the ILP (2) can be used to solve the TIMED TARGET SET SELECTION problem by ranging over different values of k.

3.3 Greedy Algorithm

We provide a greedy algorithm which finds a TTS $\mathcal{S}_0$, $\mathcal{S}_1$, $\emptyset$ for a given pair (G,τ). The algorithm first sorts the nodes in ascending order of their degrees as $v_1,\cdots,v_n$ and set $\mathcal{S}_0 = \emptyset, \mathcal{S}_1 = \emptyset$. Then, nodes are processed one by one, and they are decided to be in $\mathcal{S}_0$, in $\mathcal{S}_1$, or in neither of them. The processed nodes which are in neither $\mathcal{S}_0$ nor $\mathcal{S}_1$ are called *unselected* nodes. When processing a node v_i, a neighbor $u \in N(v_i)$ is said to be *blocked* if the number of unselected nodes in $N(u)$ is equal to $d(u) - \tau(u)$ (i.e., if v_i is set to be unselected, u cannot become positive in the first step when only the nodes in $\mathcal{S}_0$ are positive). Let $\texttt{blocked}[v_i]$ be the set of blocked nodes in $N(v_i)$. If $|\texttt{blocked}[v_i]| = 0$, we can safely set v_i to be unselected. If $|\texttt{blocked}[v_i]| > 1$, then we set v_i to be in $\mathcal{S}_0$. If

Algorithm 1: Greedy Algorithm for TTS

Sort the nodes of G in ascending order of their degrees as the sequence $v_1, \ldots, v_n$.
Set $\mathcal{S}_0 = \emptyset$, $\mathcal{S}_1 = \emptyset$, and $\texttt{unselected}[v_i] = 0$, $\texttt{blocked}[v_i] = \emptyset$ for all $1 \le i \le n$.
for $i = 1$ to n **do**
  for $u \in N(v_i)$ **do**
    if $\texttt{unselected}[u] = d(u) - \tau(u)$ **then**
      add u to $\texttt{blocked}[v_i]$
  end
  if $|\texttt{blocked}[v_i]| = 0$ **then**
    for $u \in N(v_i)$ **do**
      $\texttt{unselected}[u] += 1$
    end
  if $|\texttt{blocked}[v_i]| > 1$ **then**
    Add v_i to $\mathcal{S}_0$
  if $|\texttt{blocked}[v_i]| = 1$ **then**
    for $w \in \texttt{blocked}[v_i]$ **do**
      if $d(w) > d(v_i)$ **then**
        for $u \in N(v_i)$ **do**
          $\texttt{unselected}[u] += 1$
        end
        Add w to $\mathcal{S}_1$, set $\tau(w) = 0$
      else
        Add v_i to $\mathcal{S}_0$
      end
    end
end

$|\texttt{blocked}[v_i]| = 1$, we could either add v_i to $\mathcal{S}_0$ or set v_i to be unselected. For the latter, we add the only node in $\texttt{blocked}[v_i]$, say w, to $\mathcal{S}_1$. If $d(w) > d(v_i)$, we do the latter (since w is more "influential"), otherwise we execute the former. A precise description of the algorithm is given in Algorithm 1. It is straightforward to prove that Algorithm 1 returns a TTS, using an inductive argument. For the sake of completeness, a formal proof is given in the extended version.

Our algorithm is inspired by the well-known greedy algorithm for finding a TS (cf. [24]) where there is no $\mathcal{S}_1$ and node v_i is assigned to $\mathcal{S}_0$ if $|\texttt{blocked}[v_i]| \ge 1$ and set to be unselected otherwise. The final $\mathcal{S}_0$ is a TS which make all nodes positive in one step of the non-progressive model.

We did not provide any approximation guarantee for our algorithm, but according to Theorem 6 we cannot hope for an approximation ratio better than $\mathcal{O}(2^{\log^{1-\epsilon} n})$ for any fixed $\epsilon > 0$. Furthermore, our algorithm takes advantage of timing only for one extra step, but as we observe in the next subsection this would already allow us to produce solutions close to optimal. Devising algorithms which fully take advance of timing is left to future work.

3.4 Experiments

Our experiments were carried out on an Intel Xeon E3 CPU, with 32 GB RAM, and a Linux OS and the code is in C++ and Python.

Synthetic Networks. We conducted experiments on BA (Barabási–Albert) model [7] and ER (Erdős–Rényi) model [22]. The number of nodes were set to $n = 10, 15, \cdots, 40$ and the edge parameter, in both models, was set to obtain average degree 8 (which is comparable to the average degree of the real-world networks of similar size, for example, see Karate club network [30]). For each value of n, 10 instances of the random graph (BA and ER) were generated. Then, the optimal size of a TS and TTS and the outcome of the original greedy algorithm for TS and our algorithm for TTS were computed. The optimal solutions were found using the ILP formulations and a CBC solver [25]. We considered the strict majority threshold assignment in all these experiments. The average outcome of each approach over all 10 graph instances are reported in Figure 2. (The standard deviations were very small, namely smaller than 0.7 in all cases.) Let us make the following two observations from Figure 2:

- **Observation 1.** Our proposed greedy algorithm performs very well and returns a solution very close to the optimal one in most cases.
- **Observation 2.** In most cases, the minimum size of a TTS (retuned by TTS-OPT) is strictly smaller than the minimum size of a TS (returned by TS-OPT). This confirms that the effect of timing is not limited to theoretically tailored graphs (such as the ones given in Example 1 and Theorem 1).

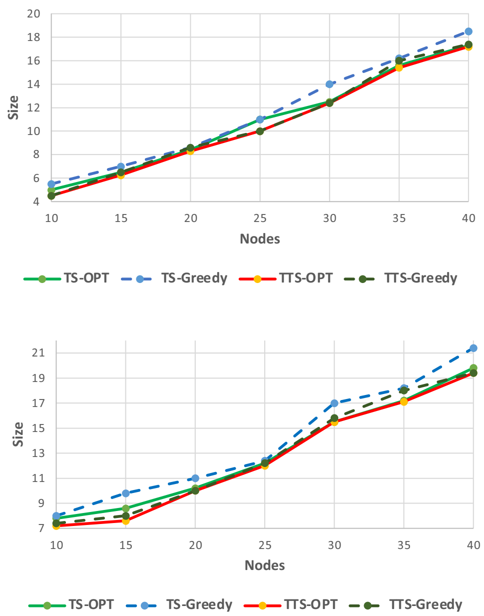

Figure 2: The average size of a TS/TTS retuned by ILP (TS-OPT and TTS-OPT) and greedy approach (TS-Greedy and TTS-Greedy) on BA graphs (top) and ER graphs (bottom).

Real-world Networks. We also have run our algorithm and the original greedy algorithm on different real-world networks, namely Twitter, Facebook, and Twitch Games from [32] where we removed edge directions for Twitter. We again used the strict majority threshold assignment. (Note that computing the optimal solutions are not possible due to the large size of networks. However, based on diagrams in Figure 2, the greedy algorithms seem to return solutions close to the optimal ones.) The outcomes, presented in Table 1, demonstrate that allowing a manipulator who conducts a greedy approach (which is the most commonly proposed mechanism in the literature for various target set selection problems, cf. [27, 24]) to target nodes at their desired time gives them more than 13% advantage for the graphs under experiment, indicating another time that timing matters, even on graph which emerge in the real world.

Network	Nodes	TTS-Greedy	TS-Greedy	Imp.
Facebook	4039	1727	1985	13%
Twitter	81306	25022	28991	13.7%
Twitch Games	168114	44795	53726	16.6%

Table 1: The size of a TS obtained by the greedy algorithm versus the size of a TTS from our greedy approach (Algorithm 1), for the strict majority, and the improvement (Imp.) in the form of (TS-TTS)/TS.

3.5 Exact Linear-time Algorithm for Trees

Our goal in this section is to provide an exact algorithm for the TIMED TARGET SET SELECTION problem for a given tree T and threshold assignment τ.

Let $\mathcal{L}(v)$ and $\bar{\mathcal{L}}(v)$ be the set of leaf and non-leaf neighbors of v, and define $l(v) := |\mathcal{L}(v)|$ and $\bar{l}(v) := |\bar{\mathcal{L}}(v)|$. Furthermore, we define $\mathcal{L}[v] := \mathcal{L}(v) \cup \{v\}$ and $\bar{\mathcal{L}}[v] := \bar{\mathcal{L}}(v) \cup \{v\}$. We partition the non-leaf nodes of T into three sets:

- $A := \{v \in V(T) : d(v) > 1, \tau(v) > \bar{l}(v)\}$
- $B := \{v \in V(T) : d(v) > 1, \tau(v) < \bar{l}(v)\}$
- $C := \{v \in V(T) : d(v) > 1, \tau(v) = \bar{l}(v)\}$.

Furthermore, we partition A into $A' := \{v \in A : \tau(v) < d(v)\}$ and $A'' := \{v \in A : \tau(v) = d(v)\}$.

Let us set a root v for T. Then, we can partition $V(T)$ into $L_0 \cup L_1 \cup \cdots \cup L_d$, where the i-th level L_i consists of nodes whose distance from v is equal to i and d is the depth of the tree. Set $L_{-1} = \emptyset$.

Let us first consider the special case of $|A'' \cup C| \leq 1$. Set the only node in $A'' \cup C$ (or an arbitrary non-leaf node if $A'' \cup C = \emptyset$) as the root. Consider the sequence $\mathcal{S}_i = (L_{d-i} \cup L_{d-i-1}) \cap A$ for $0 \leq i \leq d$. We can prove that this is a TTS by induction on d, where the base case of $d = 1$ corresponds to a star graph (similar to Example 1). Since each node in A appears exactly twice in this TTS, its size is equal to $2|A|$. On the other hand, for any TTS $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_k$ and any node $v \in A$, we have $\sum_{i=0}^{k} |\mathcal{S}_i \cap \mathcal{L}[v]| \geq 2$. This is true because if $\sum_{i=0}^{k} |\mathcal{S}_i \cap \mathcal{L}[v]| \leq 1$, then v and its leaf neighbors cannot become positive simultaneously in any step and this contradicts the assumption that $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_k$ is a TTS. (This uses that $\tau(v) > 0$ for any node v, which can be assumed as we will explain.) Since for any two distinct nodes u and v, we have $\mathcal{L}[u] \cap \mathcal{L}[v] = \emptyset$, we can conclude that $\overleftrightarrow{MTT}(T, \tau) \geq 2|A|$. Furthermore, we argued that there is a TTS of size $2|A|$. Thus, $\overleftrightarrow{MTT}(T, \tau) = 2|A|$, which explains the lines 3-5 in Algorithm 2.

Now, let's consider the case of $|A'' \cup C| \geq 2$. Again, we set a node $v \in A'' \cup C$ as the root. Then, we iterate over the nodes respectively from L_d to L_1. Let T' be the subtree induced by the node u which is being processed and its descendants. In the next two paragraphs, we explain that if T' has a certain structure, then we know how to efficiently compute the minimum size of a TTS for T' and how to connect it to the minimum size of a TTS in the original tree. This permits us to keep reducing the size of the tree while guaranteeing that the subtree induced by the next node that is processed will have one of our desired structures.

Suppose that u (not equal to the root v) is a node in A'' and none of its descendants belongs to $A'' \cup C$. Denote the parent of u by z. Let T' be the induced subtree on u and its descendants and T'' be the induced subtree on $V(T) \setminus V(T')$. For T', define $\tau'(w) = \tau(w) - 1$ if $w = u$ and $\tau'(w) = \tau(w)$ otherwise. For T'', let $\tau''(w) = \tau(w) - 1$ if $w = z$ and $\tau''(w) = \tau(w)$ otherwise. We can show that $\overleftrightarrow{MTT}(T, \tau) = \overleftrightarrow{MTT}(T', \tau') + \overleftrightarrow{MTT}(T'', \tau'')$. Using a proof similar to the case of $|A'' \cup C| \leq 1$ (which was handled before), we can prove that $\overleftrightarrow{MTT}(T', \tau') = 2|V(T') \cap A|$. (There is a similar argument for $\tau(u) = 0$.) This should explain the *if* statement in line 11.

Suppose that u is a non-root node in C and none of its descendants belongs to $A'' \cup C$. Furthermore, let T' be the induced subtree on u and its descendants and T'' be the induced subtree on $(V(T) \setminus V(T')) \cup \{u\}$. For T', we define $\tau'(w) = \tau(w) - 1$ if $w = u$ and $\tau'(w) = \tau(w)$ otherwise. For T'', we set $\tau''(w) = 1$ if $w = u$ and $\tau''(w) = \tau(w)$ otherwise. We can prove that $\overleftrightarrow{MTT}(T, \tau) = \overleftrightarrow{MTT}(T', \tau') + \overleftrightarrow{MTT}(T'', \tau'')$. Again, using an argument similar to the case of $|A'' \cup C| \leq 1$, we can prove that $\overleftrightarrow{MTT}(T', \tau') = 2|V(T') \cap A|$. This justifies the *if* statement in line 14.

In the extended version, we provide a series of lemmas which explain Algorithm 2 in a step by step and constructive fashion and prove its correctness.

Algorithm 2: Minimum Size of a TTS in Trees

1 Determine sets A, A', A'', B, C.
2 Set $x = 0$.
3 **if** $|A'' \cup C| \leq 1$ **then**
4   Set $x = 2|A|$.
5 **end**
6 **else**
7   Choose some $v \in A'' \cup C$ as the root.
8   **for** $i = d$ *to* 1 **do**
9   **for** u *in* L_i **do**
10   Let T' be the induced subtree on u and its descendants, and z be the parent of u.
11   **if** $u \in A''$ *or* $\tau(u) = 0$ **then**
12   Set $x += 2|V(T') \cap A|$. Remove $V(T')$ and set $\tau(z) = \tau(z) - 1$. Update sets A, A', A'', B, C.
13   **end**
14   **else if** $u \in C$ **then**
15   Set $x += 2|V(T') \cap A|$. Remove $V(T') \setminus \{u\}$ and set $\tau(u) = 1$. Update sets A, A', A'', B, C.
16   **end**
17   **end**
18   **end**
19   Set $x += 2|V(T) \cap A|$.
20 **end**

4 Future Work

In [8], a polynomial time algorithm for finding $\overrightarrow{MT}(G, \tau)$ on trees was provided, and it was left open to determine whether finding $\overleftrightarrow{MT}(G, \tau)$ is tractable on trees. Recently, a polynomial time algorithm was given [39] when $\tau(v) \in \{0, 1, d(v)\}$ for any node v, but the problem remains open for general threshold assignment. We provided a linear-time algorithm to compute $\overleftrightarrow{MTT}(G, \tau)$ on trees for any threshold assignment. Can our techniques be leveraged to settle the problem for $\overleftrightarrow{MT}(G, \tau)$?

In practice, targeting some nodes, such as "influencers", might be more costly than others. A potential avenue for future research is to study the set-up where each node has a cost assigned to it and the manipulator aims to minimize the cost.

Finally, it would be interesting to devise new algorithms which take advantage of timing fully (unlike our greedy approach) and evaluate their performance on different real-world data.

References

[1] Eyal Ackerman, Oren Ben-Zwi, and Guy Wolfovitz, 'Combinatorial model and bounds for target set selection', *Theoretical Computer Science*, **411**(44-46), 4017–4022, (2010).

[2] Sarah Spence Adams, Denise Sakai Troxell, and S Luke Zinnen, 'Dynamic monopolies and feedback vertex sets in hexagonal grids', *Computers & Mathematics with Applications*, **62**(11), 4049–4057, (2011).

[3] Charu C Aggarwal, Shuyang Lin, and Philip S Yu, 'On influential node discovery in dynamic social networks', in *Proceedings of the 2012 SIAM International Conference on Data Mining*, pp. 636–647. SIAM, (2012).

[4] Michael Aizenman and Joel L Lebowitz, 'Metastability effects in bootstrap percolation', *Journal of Physics A: Mathematical and General*, **21**(19), 3801, (1988).

[5] Aris Anagnostopoulos, Luca Becchetti, Emilio Cruciani, Francesco Pasquale, and Sarai Rizzo, 'Biased opinion dynamics: When the devil is in the details', in *IJCAI*, (2020).

[6] Vincenzo Auletta, Diodato Ferraioli, and Gianluigi Greco, 'Reasoning about consensus when opinions diffuse through majority dynamics', in *IJCAI*, (2018).

[7] Albert-László Barabási and Réka Albert, 'Emergence of scaling in random networks', *science*, **286**(5439), 509–512, (1999).

[8] Oren Ben-Zwi, Danny Hermelin, Daniel Lokshtanov, and Ilan Newman, 'Treewidth governs the complexity of target set selection', *Discrete Optimization*, **8**(1), 87–96, (2011).

[9] Robert Bredereck and Edith Elkind, 'Manipulating opinion diffusion in social networks', in *IJCAI*, (2017).

[10] Robert Bredereck, Lilian Jacobs, and Leon Kellerhals, 'Maximizing the spread of an opinion in few steps: opinion diffusion in non-binary networks', in *IJCAI*, pp. 1622–1628, (2021).

[11] Markus Brill, Edith Elkind, Ulle Endriss, and Umberto Grandi, 'Pairwise diffusion of preference rankings in social networks', in *IJCAI*, (2016).

[12] Matteo Castiglioni, Diodato Ferraioli, and Nicola Gatti, 'Election control in social networks via edge addition or removal', in *AAAI*, (2020).

[13] Carmen C Centeno, Mitre C Dourado, Lucia Draque Penso, Dieter Rautenbach, and Jayme L Szwarcfiter, 'Irreversible conversion of graphs', *Theoretical Computer Science*, **412**(29), 3693–3700, (2011).

[14] Cheng-Lung Chen, Eduardo L Pasiliao, and Vladimir Boginski, 'A polyhedral approach to least cost influence maximization in social networks', *Journal of Combinatorial Optimization*, **45**(1), 1–31, (2023).

[15] Ning Chen, 'On the approximability of influence in social networks', *SIAM Journal on Discrete Mathematics*, **23**(3), 1400–1415, (2009).

[16] Wei Chen, Wei Lu, and Ning Zhang, 'Time-critical influence maximization in social networks with time-delayed diffusion process', in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pp. 591–598, (2012).

[17] Chun-Ying Chiang, Liang-Hao Huang, Bo Li Jr, Jiaojiao Wu, and Hong-Gwa Yeh, 'Some results on the target set selection problem', *Journal of Combinatorial Optimization*, **25**(4), 702–715, (2013).

[18] Dmitry Chistikov, Grzegorz Lisowski, Mike Paterson, and Paolo Turrini, 'Convergence of opinion diffusion is pspace-complete', in *AAAI*, volume 34, pp. 7103–7110, (2020).

[19] Amin Coja-Oghlan, Uriel Feige, Michael Krivelevich, and Daniel Reichman, 'Contagious sets in expanders', in *SODA*, pp. 1953–1987, (2014).

[20] F Corò, E Cruciani, G D'Angelo, and S Ponziani, 'Exploiting social influence to control elections based on scoring rules', in *IJCAI*, volume 1, pp. 201–207, (2019).

[21] Mitre C Dourado, Lucia Draque Penso, Dieter Rautenbach, and Jayme L Szwarcfiter, 'Reversible iterative graph processes', *Theoretical Computer Science*, **460**, 16–25, (2012).

[22] László Erdős, Antti Knowles, Horng-Tzer Yau, and Jun Yin, 'Spectral statistics of erdős–rényi graphs i: Local semicircle law', (2013).

[23] Piotr Faliszewski, Rica Gonen, Martin Koutecký, and Nimrod Talmon, 'Opinion diffusion and campaigning on society graphs', in *IJCAI*, (2018).

[24] MohammadAmin Fazli, Mohammad Ghodsi, Jafar Habibi, Pooya Jalaly, Vahab Mirrokni, and Sina Sadeghian, 'On non-progressive spread of influence through social networks', *Theoretical Computer Science*, **550**, 36–50, (2014).

[25] John Forrest and Robin Lougee-Heimer, 'Cbc user guide', in *Emerging theory, methods, and applications*, 257–277, INFORMS, (2005).

[26] Eric Goles and Jorge Olivos, 'Periodic behaviour of generalized threshold functions', *Discrete mathematics*, **30**(2), 187–189, (1980).

[27] David Kempe, Jon Kleinberg, and Éva Tardos, 'Maximizing the spread of influence through a social network', in *SIGKDD*, pp. 137–146, (2003).

[28] Kaveh Khoshkhah, Hossein Soltani, and Manouchehr Zaker, 'Dynamic monopolies in directed graphs: the spread of unilateral influence in social networks', *Discrete Applied Mathematics*, **171**, 81–89, (2014).

[29] Jinha Kim, Wonyeol Lee, and Hwanjo Yu, 'Ct-ic: Continuously activated and time-restricted independent cascade model for viral marketing', *Knowledge-Based Systems*, **62**, 57–68, (2014).

[30] Jérôme Kunegis, 'Konect: the koblenz network collection', in *Proceedings of the 22nd international conference on world wide web*, pp. 1343–1350, (2013).

[31] Hicham Lesfari, Frédéric Giroire, and Stéphane Pérennes, 'Biased majority opinion dynamics: Exploiting graph k-domination', in *IJCAI*, (2022).

[32] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[33] S Mishra, Jaikumar Radhakrishnan, and Sivaramakrishnan Sivasubramanian, 'On the hardness of approximating minimum monopoly problems', in *FSTTCS*, pp. 277–288, (2002).

[34] Robert Morris, 'Minimal percolating sets in bootstrap percolation.', *The Electronic Journal of Combinatorics [electronic only]*, **16**(1), Research–Paper, (2009).

[35] Elchanan Mossel, Joe Neeman, and Omer Tamuz, 'Majority dynamics and aggregation of information in social networks', *AAMAS*, **28**(3), 408–429, (2014).

[36] André Nichterlein, Rolf Niedermeier, Johannes Uhlmann, and Mathias Weller, 'On tractable cases of target set selection', *Social Network Analysis and Mining*, **3**(2), 233–256, (2013).

[37] David Peleg, 'Size bounds for dynamic monopolies', *Discrete Applied Mathematics*, **86**(2-3), 263–273, (1998).

[38] Svatopluk Poljak and Daniel Turzík, 'On pre-periods of discrete influence systems', *Discrete Applied Mathematics*, **13**(1), 33–39, (1986).

[39] Dieter Rautenbach, Stefan Ehard, and Julien Baste, 'Non-monotone target sets for threshold values restricted to 0, 1, and the vertex degree', *Discrete Mathematics & Theoretical Computer Science*, **24**, (2022).

[40] Hossein Soltani and Babak Moazzez, 'A polyhedral study of dynamic monopolies', *Annals of Operations Research*, **279**(1), 71–87, (2019).

[41] Hossein Soltani, Ahad N Zehmakan, and Ataabak B Hushmandi, 'Minimum target sets in non-progressive threshold models: When timing matters', *arXiv preprint arXiv:2307.09111*, (2023).

[42] Ahad N Zehmakan, 'Opinion forming in erdős–rényi random graph and expanders', *Discrete Applied Mathematics*, **277**, 280–290, (2020).

[43] Ahad N Zehmakan, 'Majority opinion diffusion in social networks: An adversarial approach', in *AAAI*, volume 35, pp. 5611–5619, (2021).

[44] Ahad N Zehmakan, 'Random majority opinion diffusion: Stabilization time, absorbing states, and influential nodes', *arXiv preprint arXiv:2302.06760*, (2023).

[45] Huiyuan Zhang, Thang N Dinh, and My T Thai, 'Maximizing the spread of positive influence in online social networks', in *2013 IEEE 33rd International Conference on Distributed Computing Systems*, pp. 317–326. IEEE, (2013).