Node Conversion Optimization in Multi-hop Influence Networks

Jie Zhang
University of Bath
Bath, United Kingdom
jz2558@bath.ac.uk

Yuezhou Lv
Impeccable Tech Pte. Ltd.
Singapore, Singapore
lvyuezhou@gmail.com

Zihe Wang*
Renmin University of China
Beijing, China
wang.zihe@ruc.edu.cn

ABSTRACT

In this paper, we study scenarios such as diffusion of innovations in a social system and belief propagation in social choice decision-making, which can be captured by a social influence network. In such networks, nodes are distributed and are connected by links between them. Nodes have two different states, s and r. They can change from state s to state r, but not backward [24]. Nodes are interested in changing to state r only if a sufficient number of their neighbors change to state r. In many scenarios, it is desired to design *local decision algorithms* that guarantee this feature, termed as the *safety* of node conversion.

We design optimal algorithms that maximize the number of nodes that change to state r. In particular, we assume that each node can observe its neighbors up to a distance of k from itself, which introduces complexity to the setting that each node can only observe its immediate neighbors, i.e., $k = 1$. Moreover, we consider the models that nodes have the same threshold or different thresholds under which their conversion from s to r is safe. We first present the optimal algorithm for the uniform threshold model and establish its optimality by characterizing a monotonicity property. We then generalize the algorithm to maximize node conversion when they have different threshold values. The monotonicity properties and insights on nodes' recursive reasoning of their neighbors' status may be of independent interest.

KEYWORDS

Influence network, local decision, multi-hop, node conversion optimization.

ACM Reference Format:
Jie Zhang, Yuezhou Lv, and Zihe Wang*. 2023. Node Conversion Optimization in Multi-hop Influence Networks. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 8 pages.

1 INTRODUCTION

In this paper, we follow the literature [4, 8, 18, 32] and consider the distributed decision-making multi-agent coordination problem. In daily life, most decisions people make, from which new products to buy to whom to vote for, are influenced by their friends' choices. Taking the diffusion of innovations, for example, often, individuals only wish to buy a new product if a sufficient fraction of their friends do. The purchase from their friends justifies the popularity and quality of the new product. For another example, when the shareholders of a company defend against a hostile takeover, an essential factor in their decision-making is how many other shareholders are resisting the acquisition. Naturally, the more shares have been acquired by the bidder, the more likely they would sell their shares. This empirical evidence on these peer effects motivates the theoretical study of Influence Networks.

On an influence network, agents are spatially separated but are linked by mutual relationships. Although communication is a powerful instrument to enable coordination, sometimes it is not as efficient as it is supposed to be [9, 10]. Thereafter, intelligent agents must be able to optimize their actions even without the benefit of communication. In many practical applications, indeed, communication may not be feasible. For instance, embedded platforms have strict energy constraints; communication channels are under electromagnetic interference; enemies can detect security risks of potential interception of messages in hostile territory. Therefore, it is natural to ask the question that can agents coordinate without communication? We tackle this question by considering situations where communication is impossible or forbidden, popularized by [3, 13, 16].

We consider a model in which agents have two possible statuses and can observe the network's local topology. Each agent can either keep its current status (not buying a new product and not attending a conference) or change to a different status (buying a new product and attending a conference). They would like to change their status if there are a sufficient number of their peers who do so; otherwise, they prefer to keep their original status. To prevent the system from being stuck in a suboptimal configuration, we are interested in protocols that change as many agents' statuses as possible.

Existing literature in studying influence networks has been implicitly assumed that not only does the individual's preferences depend on the status of their neighbors, but also that each individual only knows the information of their immediate neighbors. Therefore, agents do not mull over information beyond its 1-hop neighbors. This way, problems such as determining conditions under which it is safe to change status and maximizing node conversion is often tractable. However, in many scenarios, such as the aforementioned example in which academics decide to attend a conference, their decision is affected by their close colleagues and a wider range of friends who will attend the conference. In addition, each individual could have a different threshold on their neighbors' conversion beyond which they will decide to attend the conference. Hence, investigating the multi-hop information setting and generalizing the uniform threshold case to allow agents to have arbitrary thresholds is a natural extension in studying influence networks. In this case, on the one hand, the agents stand a better chance to make their decision if they view further on these networks; on the other hand, enabling agents to have distant views introduces extra complexity in their decision-making analysis.

In this paper, we allow agents to perceive their k-hop neighbors and their thresholds. We design algorithms to facilitate agents'

Zihe Wang is the corresponding author.

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

distributed decision-making. We present optimal algorithms to maximize the number of agents changing status in any influence network on the premise that the transition is safe. Moreover, we extend previous works by generalizing the model on another aspect. That is, to allow agents having different thresholds. Our analyses show that the general threshold model and the k-hop setting fundamentally change how the system behaves.

Due to the nature of the setting, our algorithms recursively reason for a node's neighbors' possible actions. To make sure that the transition is safe, an agent needs to view and reason through its neighbors' lens, and the neighbors do the same while making their decisions. This ponderation naturally introduces *a hierarchy of views*, which lies underneath the design of our algorithms. To maximize the number of nodes conversion, we introduce a two-phase "prediction-counting" approach for designing *local decision algorithms*. In the *prediction* phase, we classify the neighbors of an agent into different groups according to whether their views are the same as the agent's view or their views are strictly smaller than its view. The agent then *conservatively* predicts its neighbors' decisions. This is because even if there might be extra information beyond the agent's perception that would trigger its neighbors' conversion, the agent cannot take the risk. This way, we narrow down the decision-making problem to a smaller scale, which becomes tractable in the *counting* phase. En route to show the optimality of the algorithms, we peek behind the hierarchy-of-views curtain by characterizing its monotonicity properties.

1.1 Related Work

The influence network model presented in this paper is motivated by [5, 6, 11, 19, 22–24] and their variants. Models allowing nodes to convert from state A to state B but not backward have been widely studied, to name a few, see [27, 36, 41]. The threshold effects in these models are discussed in [25, 40]. In which, the most relevant work to ours is [24]. In the paper, the authors define a hierarchy of safety properties and devise algorithms for maximizing node conversion for each hierarchy. When constrained by the strongest safety guarantee, under which the class of algorithms is fully contained in the class of lower safety levels, their algorithm is optimal. However, for lower safety level guarantees that admit a larger class of algorithms, their algorithms are not optimal unless additional assumptions on the network structure are made. In addition, their algorithms include an exhaustive search procedure that is impractical. Since then, the problem of devising more constructive algorithms that are applicable to any network remains open. We improve the state-of-the-art by devising optimal protocol-safe algorithms. We present algorithms that work for the uniform threshold case and the case of the general threshold.

The influence network is a typical Collective Action model which studies the actions take by a group of agents to achieve their common goal. These models have been widely applied in biology [14, 26], economics [15, 20, 35], robotics [33], and sociology [1, 12, 30].

In scenarios where communication is feasible, it plays an important role in facilitating coordination. For example, communication is used to organize all loyal generals to agree upon a battle plan [21], reach an agreement in distributed systems [31], or improve players' performance in tasks [7, 13]. When communication is not feasible, one circumvention is letting agents leverage some public information to coordinate. For example, in the standard beauty contest game [28] and price competition in a duopoly [29], agents have access to a public signal which drastically reduces the expected behavior uncertainty and thus leads to coordination. Another circumvention is that agents can make use of the observable historical activity they produce to generate cooperative behaviors [2, 34, 37, 38]. In this paper, though, we study a one-shot model in which there is no historical data, and the only information available for the agents is their observed network structure.

In the game theory domain, [19] provides an overview of the literature analyzing games where players are connected via a network structure. [22] model a public goods game played on a social network. In [39] and [17], the authors employ propositional logic to study the incentive engineering problems in Boolean games. Together with other coalitional game theory models, these works concern the typical questions in game theory, such as equilibrium existence and the complexity of best response dynamics.

2 PRELIMINARIES

We model an Influence Network as a connected undirected simple graph $G = (V, E)$ in which V denotes a set of nodes, E denotes a set of (undirected) edges. In a *k-hop influence network*, each node $v \in V$ has complete information about the topology of the network up to a distance of k from itself, known as its k-hop neighborhood. It includes any node u that is connected to v by a path of at most k edges and the edges on this path. Note that it includes node v itself as well. Let $V_k(v)$ and $E_k(v)$ denote the set of nodes and edges within the k-hop neighborhood of node v, respectively. There are two possible states of a node, s and r. The nodes can convert from state s to state r, but they cannot convert back to s. Without loss of generality[1], we assume that all nodes start with state s. We denote the nodes' binary decision on their states as a function $d : V \to \{r, s\}$. Denote $\Phi : V \to Z^+$ a threshold function. That is, for each node v, if there are at least $\Phi(v)$ number of nodes, including itself, in its k-hop neighborhood ending up in state r, then $d(v) = r$ is a *safe* action for node v; otherwise, $d(v) = s$ is safe. We define the *k-hop view* and a *view* of a node as follows.

DEFINITION 2.1. *A* view $\Gamma(v)$ *of node* v *is* $\Gamma(v) = (V(v), E(v))$, *where* $V(v) \subseteq V_k(v), E(v) \subseteq E_k(v)$. *In particular, the* k-hop view $\Gamma_k(v)$ *of node* v *is* $\Gamma_k(v) = (V_k(v), E_k(v))$.

We note that the threshold of a node is an invariant in any view. Node v has access to the value $\Phi(u)$, for any node u within its k-hop view, i.e., $u \in V_k(v)$. For any vertex set S, let $\Phi(S)$ denote the maximum threshold of nodes in S. That is, $\Phi(S) = \max_{v \in S}\{\Phi(v)\}$. Throughout the paper, with a slight abuse of notation, we may denote a node u in node v's view by $u \in \Gamma(v)$. That is, $u \in V(v)$ and $u \in \Gamma(v)$ are used interchangeably. Depending on whether a view is a proper subset of another view, we define the *equivalent view* and the *sub-view*, respectively.

- *Equivalent view*: View $\Gamma(v)$ is equivalent to view $\Gamma(u)$, denoted by $\Gamma(v) = \Gamma(u)$, if $V(v) = V(u), E(v) = E(u)$.

[1] Even if some nodes start with state r, they do not introduce any uncertainty when their neighbors reason about their possible actions, as they cannot convert back to s.

- *Sub-view*: View $\Gamma(v)$ is a sub-view of view $\Gamma(u)$, denoted by $\Gamma(v) \subseteq \Gamma(u)$, if $V(v) \subseteq V(u), E(v) \subseteq E(u)$, and $\Gamma(v) \neq \Gamma(u)$. That is, $\Gamma(v)$ can be obtained by removing some nodes or edges from $\Gamma(u)$.

A *local decision algorithm* $\mathcal{F}$ is a distributed protocol designed by some global algorithm designer. It takes each node v and its view $\Gamma_k(v)$ as input, and outputs a decision $d(v)$, i.e., $d(v) = \mathcal{F}(v, \Gamma_k(v))$. We are interested in algorithms that are *protocol-safe*.

DEFINITION 2.2. *An algorithm $\mathcal{F}$ is* protocol-safe *if it outputs a safe decision $d(v)$ for any node v and any influence network G.*

Within the class of protocol-safe algorithms, we are particularly interested in the algorithm that converts as many nodes to state r as possible, in any network. That is, the *optimal* algorithm. Denote $|S|$ the cardinality of set S.

DEFINITION 2.3. *A protocol-safe algorithm $\mathcal{F}^*$ is* optimal *if $|\{v \in V \mid \mathcal{F}^*(v, \Gamma(v)) = r\}| \geq |\{v \in V \mid \mathcal{F}(v, \Gamma(v)) = r\}|$, for any other safe algorithm $\mathcal{F}$ and network G.*

In designing the optimal algorithms, it is unavoidable that a node v needs to recursively view the network through another node u's lens subject to its own k-hop neighborhood. Node v does so to infer whether, given this information, node u will convert to r. For this reason, we define *visualization* $\Gamma(v; u)$ for ease of notation.

DEFINITION 2.4. $\Gamma(v; u)$ *is node v's* visualization *of node u's view, obtained by removing all nodes in $\Gamma(v)$ whose distance to u is larger than k, as well as all edges that are incident to those nodes.*

In other words, $\Gamma(v; u)$ is the part of u's view that node v can perceive, subject to v's scope. By definition, $\Gamma(v; u) \subset \Gamma(v)$. However, we note that a visualization is not equivalent to the intersection of two views, i.e., $\Gamma(v; u) \neq \Gamma(v) \cap \Gamma(u)$. We present such an example in Figure 1.

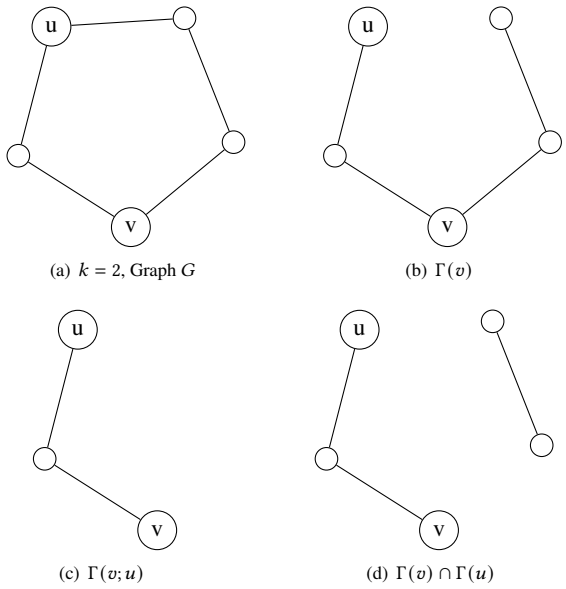
We note that *visualization* can be recursively defined. For example, the visualization $\Gamma(v; u; w)$ is obtained by removing all nodes in $\Gamma(v; u)$ whose distance to w is larger than k, as well as all edges incident to those nodes. In case that v and w are the same node, $\Gamma(v; u; w) = \Gamma(v; u)$.

3 THE HEURISTICS AND INTUITION

A heuristic towards designing the optimal algorithm is to identify a set of nodes that are within each other's view, and such that the number of these nodes is larger than or equal to their threshold values. If these nodes collectively convert from state s to state r, their conversion is safe. We formalize this idea by defining an *eligible subgraph*, which will be used throughout the paper.

DEFINITION 3.1. *A non-empty subgraph S_E of the network is* eligible *if*
(i) The diameter of S_E is at most k, i.e., the greatest distance between any pair of nodes in S_E is at most k.
(ii) For any node $v \in V(S_E)$, $S_E \subseteq \Gamma(v)$, i.e., every node and edge of S_E is within node v's view.
(iii) $|V(S_E)| \geq \Phi(S_E)$. That is, the cardinality of set S_E is no less than the maximum threshold of nodes in set $V(S_E)$.

The first two conditions guarantee that the nodes in $V(S_E)$ are within each other's view. The third condition guarantees that there are a sufficient number of nodes such that their conversion to state r is safe.



(a) $k = 2$, Graph G (b) $\Gamma(v)$ (c) $\Gamma(v; u)$ (d) $\Gamma(v) \cap \Gamma(u)$

Figure 1: $\Gamma(v; u)$ is not necessarily the same as $\Gamma(v) \cap \Gamma(u)$

Following this idea, a heuristic algorithm is to search for eligible subgraphs of the network. It is protocol-safe, but not optimal, which is evident in the following example.

EXAMPLE 1. *Consider a 2-hop influence network that is a path with five nodes v_1, v_2, v_3, v_4, and v_5. Their threshold values are $\Phi(v_1) = \Phi(v_2) = \Phi(v_4) = \Phi(v_5) = 2$, and $\Phi(v_3) = 5$. For any heuristic algorithms that search for an eligible subgraph, it will not include the central node v_3, as the threshold $\Phi(v_3) = 5$ is greater than the cardinality of any subgraph with a diameter of at most 2. Therefore, v_3 will remain in state s. So, these algorithms can only convert at most four nodes to state r. Nevertheless, as we will see later, the optimal algorithm $\mathcal{F}^*_G$ introduced in Section 5 will convert all five nodes to state r.*

Before introducing the optimal algorithm, we provide a high-level intuition in its design, which follows a maximin principle. On the one hand, for an algorithm to be safe, we have to consider the worst scenario. That is, beyond node v's visualization of a node u's view, we assume that there are no extra nodes to trigger node u's conversion to state r, which may effectively trigger v's conversion to state r being a safe action. To this end, we ignore the nodes that are beyond v's k-hop neighborhood. On the other hand, to maximize the number of nodes that would convert to state r on any graph, we shall make full use of the nodes within v's k-hop view. To this end, we introduce a two-phase approach.

In the Prediction phase, we recursively examine smaller views until it boils down to a stage that either the view of the considering node is equivalent to its upper-layer node's view, or it is a sub-view of its upper-layer node's view and the node contains a sufficient number of nodes within its view to secure its conversion to r. We carry over these two types of nodes to the Counting Phase and disregard other

indefinite nodes. In the Counting Phase, we convert as many nodes as possible to r on the premise that it is safe. Depending on whether all nodes have the same threshold value, the exact execution of this process differs. The optimality of the algorithms is built upon a monotonicity property.

4 THE UNIFORM THRESHOLD MODEL

In this section, we consider the case that all nodes have the same threshold value Φ. We present the algorithm $\mathcal{F}_U^*$ as follows and show that it is optimal in this model.

Algorithm 1 Algorithm $\mathcal{F}_U^*(v, \Gamma(v))$

$S_{sub} = S_{eql} = \emptyset$. \\ local variables
–The Prediction Phase–
for each node u in $\Gamma(v)$ **do**
  if $\Gamma(v;u) \subsetneq \Gamma(v)$ **then**
    if $\mathcal{F}_U^*(u, \Gamma(v;u)) = r$ **then**
      $S_{sub} \leftarrow S_{sub} \cup \{u\}$.
    end if
  else
    $S_{eql} \leftarrow S_{eql} \cup \{u\}$.
  end if
end for
–The Counting Phase–
if $|S_{sub}| + |S_{eql}| \geq \Phi(v)$ **then**
  return r.
else
  return s.
end if

The algorithm $\mathcal{F}_U^*(v, \Gamma(v))$ takes a node v and its view as input. It classifies the nodes in $\Gamma(v)$ into three sets, i.e., S_{eql}, S_{sub}, and $V(v) \setminus (S_{sub} \cup S_{eql})$. S_{eql} contains those nodes u such that the visualization $\Gamma(v;u)$ is equivalent to $\Gamma(v)$; S_{sub} contains those nodes u such that $\Gamma(v;u)$ is a sub-view of $\Gamma(v)$, and when the algorithm proceeds with taking u and the visualization $\Gamma(v;u)$ as inputs, eventually the local information is sufficient to assure that node u's conversion to r is safe. Nodes in the set $V(v) \setminus (S_{sub} \cup S_{eql})$ are those that would not necessarily convert to r, subject to v's view. That is, there might be some nodes beyond v's view $\Gamma(v)$ such that, in the event that they convert to r, it would be safe for nodes in $V(v) \setminus (S_{sub} \cup S_{eql})$ to convert to r; as a consequence, it would be safe for v to convert to r as well. However, as node v can observe k-hop local information only, it is not safe to account on these nodes to convert to r. Therefore, following a risk-averse reasoning, node v would ignore the nodes in $V(v) \setminus (S_{sub} \cup S_{eql})$. In contrast, the state of nodes in the set S_{sub} is settled as they are committed to converting to r with the proceeding of the algorithm. Nodes in S_{eql} are more accessible to v as these nodes' views are equivalent to its view, through the lens of v itself. By counting on the conversion of nodes in $S_{sub} \cup S_{eql}$, the algorithm converts as many nodes to state r as possible.

We note that the algorithm guarantees that a node $u \in \Gamma(v)$, which converts to r according to subproblem $\mathcal{F}_U^*(u, \Gamma(v;u)) = r$, is assured to convert to r when implementing $\mathcal{F}_U^*(u, \Gamma(u))$. This is critical in proving that the algorithm is protocol-safe. Since $\Gamma(v;u) \subseteq \Gamma(u)$, we denote this fact as the *monotonicity* property of an algorithm.

DEFINITION 4.1. *An algorithm $\mathcal{F}$ is monotone if for any node v, a view $\Gamma(v)$ and its sub-view $\Gamma'(v) \subseteq \Gamma(v)$, the fact that $\mathcal{F}(v, \Gamma'(v)) = r$ implies $\mathcal{F}(v, \Gamma(v)) = r$.*

LEMMA 4.1. *$\mathcal{F}_U^*$ is monotone.*

PROOF. Assume by contradiction that $\mathcal{F}_U^*$ is not monotone, then there must exist a node v, a view $\Gamma(v)$ and its sub-view $\Gamma'(v) \subsetneq \Gamma(v)$, such that $\mathcal{F}_U^*(v, \Gamma'(v)) = r$ and $\mathcal{F}_U^*(v, \Gamma(v)) = s$. Amongst these cases, take the one such that the number of nodes in view $\Gamma(v)$ is the smallest. If there are multiple such views, take the one with the least number of edges. If there is still a tie, we break ties arbitrarily.

Denote by S_{sub} and S_{eql} the corresponding sets when the algorithm $\mathcal{F}_U^*$ takes v and the view $\Gamma(v)$ as input; denote by S'_{sub} and S'_{eql} the corresponding sets when the algorithm $\mathcal{F}_U^*$ takes v and the sub-view $\Gamma'(v)$ as input. Since $\mathcal{F}_U^*(v, \Gamma'(v)) = r$, we have that

$$|S'_{sub}| + |S'_{eql}| \geq \Phi, \tag{1}$$

$$\mathcal{F}_U^*(u, \Gamma'(v;u)) = r, \quad \forall u \in S'_{sub}, \tag{2}$$

$$\mathcal{F}_U^*(u, \Gamma'(v;u)) = r, \quad \forall u \in S'_{eql}. \tag{3}$$

Combining (2) and (3), we obtain that

$$\mathcal{F}_U^*(u, \Gamma'(v;u)) = r, \quad \forall u \in S'_{sub} \cup S'_{eql}.$$

In addition, according to our choice of a view $\Gamma(v)$ with the least number of nodes and secondly the least number of edges, we get that for any $u \in S'_{sub} \cup S'_{eql}$, it holds that the node u in $\Gamma(v)$ satisfies either

- $\Gamma(v;u) = \Gamma(v)$, or
- $\Gamma(v;u) \subsetneq \Gamma(v)$ and $\mathcal{F}_U^*(u, \Gamma(v;u)) = r$.

Otherwise, if $\Gamma(v;u) \subsetneq \Gamma(v)$ and $\mathcal{F}_U^*(u, \Gamma(v;u)) = s$, we can infer that $\mathcal{F}_U^*(u, \Gamma'(v;u)) = s$ which contradicts the assumption. So, $u \in S'_{sub} \cup S'_{eql}$ implies $u \in S_{sub} \cup S_{eql}$. Therefore,

$$|S_{sub}| + |S_{eql}| \geq |S'_{sub}| + |S'_{eql}| \geq \Phi,$$

which implies that $\mathcal{F}_U^*(v, \Gamma(v)) = r$. A contradiction occurs. □

With the monotonicity property, we show that the algorithm is protocol-safe.

THEOREM 4.1. *$\mathcal{F}_U^*$ is protocol-safe.*

PROOF. To show that $\mathcal{F}_U^*$ is safe, we need to show that at least Φ nodes within v's k-hop neighborhood converted to state r, if $\mathcal{F}_U^*(v, \Gamma(v)) = r$. Since $|S_{sub}| + |S_{eql}| \geq \Phi$, this can be guaranteed if nodes $u \in \Gamma(v)$ in the sets S_{sub} and S_{eql} will indeed convert to r when the algorithm takes them and their views as inputs. For a node $u \in S_{sub}$, according to the monotonicity property, $\mathcal{F}_U^*(u, \Gamma(v;u)) = r$ implies $\mathcal{F}_U^*(u, \Gamma(u)) = r$, as $\Gamma(v;u) \subseteq \Gamma(u)$. For a node $u \in S_{eql}$, $\Gamma(v;u) = \Gamma(v)$ implies $\Gamma(v) \subseteq \Gamma(u)$. In case $\Gamma(v) = \Gamma(u)$, then $\mathcal{F}_U^*(v, \Gamma(v)) = r$ implies that $\mathcal{F}_U^*(u, \Gamma(v)) = r$, and further implies that $\mathcal{F}_U^*(u, \Gamma(u)) = r$; in case $\Gamma(v) \subsetneq \Gamma(u)$, then $\mathcal{F}_U^*(u, \Gamma(v)) = r$ implies that $\mathcal{F}_U^*(u, \Gamma(u)) = r$ due to the monotonicity property. So, $\mathcal{F}_U^*(u, \Gamma(u)) = r, \forall u \in S_{sub} \cup S_{eql}$. □

Next, we show the first main result.

THEOREM 4.2. *$\mathcal{F}_U^*$ is optimal when the nodes have the same threshold value.*

PROOF. We prove the optimality of $\mathcal{F}_U^*$ by contradiction. Suppose there is a safe algorithm $\mathcal{F}'$, a node v and its view $\Gamma(v)$, such that $\mathcal{F}'(v, \Gamma(v)) = r$ and $\mathcal{F}_U^*(v, \Gamma(v)) = s$. Furthermore, we assume that $\Gamma'(v)$ is the view with the minimum number of nodes and, in case of a tie, with a minimum number of edges such that $\mathcal{F}'(v, \Gamma'(v)) = r$ and $\mathcal{F}_U^*(v, \Gamma'(v)) = s$. Therefore, by executing algorithm $\mathcal{F}_U^*(v, \Gamma'(v))$, we know that $|S_{sub}| + |S_{eql}| < \Phi$ where S_{sub} contains any node $u \in \Gamma'(v)$ such that 1) $\Gamma'(v; u) \neq \Gamma'(v)$ and 2) $\mathcal{F}_U^*(u, \Gamma'(v; u)) = r$, and S_{eql} contains any node $u \in \Gamma'(v)$ that $\Gamma'(v; u) = \Gamma'(v)$. Denote S_{nsub} the set of nodes in $\Gamma'(v)$ but not in $S_{sub} \cup S_{eql}$. We can infer that for any $u \in S_{nsub}$, it holds that 1) $\Gamma'(v; u) \neq \Gamma'(v)$ and 2) $\mathcal{F}_U^*(u, \Gamma'(v; u)) = s$. According to the assumption, we can infer that for any $u \in S_{nsub}$, $\mathcal{F}'(u, \Gamma'(v; u)) = s$. Otherwise, as $\Gamma'(v; u) \neq \Gamma'(v)$, $\Gamma'(v; u)$ either has fewer nodes than $\Gamma'(v)$ or has the same number of nodes but fewer edges than $\Gamma'(v)$ and satisfies $\mathcal{F}_U^*(u, \Gamma'(v; u)) = s$ and $\mathcal{F}'(u, \Gamma'(v; u)) = r$, which contradicts the assumption. Then the total number of nodes that convert to state r is no more than $|S_{sub}| + |S_{eql}| < \Phi$, which contradicts the assumption that $\mathcal{F}'$ is safe. □

5 THE GENERAL THRESHOLD MODEL

In this section, we generalize the optimal algorithm $\mathcal{F}_U^*$ to allow the nodes having different threshold values. Since different nodes have different thresholds, after classifying the nodes in $\Gamma(v)$ into S_{eql} and S_{sub}, we adapt the Counting Phase in $\mathcal{F}_U^*$ so that the optimality is preserved.

Algorithm 2 Algorithm $\mathcal{F}_G^*(v, \Gamma(v))$

$S_{sub} = S_{eql} = S_{cad} = S_{max} = \emptyset$.

–The Prediction Phase–

1: **for** each node u in $\Gamma(v)$ **do**
2: **if** $\Gamma(v; u) \subsetneq \Gamma(v)$ **then**
3: **if** $\mathcal{F}_G^*(u, \Gamma(v; u)) = r$ **then**
4: $S_{sub} \leftarrow S_{sub} \cup \{u\}$.
5: **end if**
6: **else**
7: $S_{eql} \leftarrow S_{eql} \cup \{u\}$.
8: **end if**
9: **end for**

–The Counting Phase–

10: **for** $i = |S_{sub}| + |S_{eql}|, ..., 1, 0$ **do**
11: $S_{cad} \leftarrow \{u \mid u \in S_{eql}$ *and* $\Phi(u) \leq i\}$.
12: **if** $|S_{sub}| + |S_{cad}| \geq i$ **then**
13: $S_{max} \leftarrow S_{cad}$, {//used in the proof}
14: **if** $i \geq \Phi(v)$ **then**
15: return r.
16: **else**
17: return s.
18: **end if**
19: **end if**
20: **end for**

We run Algorithm 2 on Example 1, to check whether it will safely convert all five nodes to r. For node v_1, it is clear that $v_1, v_2, v_3 \in \Gamma(v_1)$. In addition,

$$\Gamma(v_1; v_2) = \Gamma(v_1) \quad \text{and} \quad \Gamma(v_1; v_3) = \Gamma(v_1).$$

So, $v_1, v_2, v_3 \in S_{eql}$ and $S_{sub} = \emptyset$. In the Counting Phase, for $i = |S_{sub}| + |S_{eql}| = 3$, $S_{cad} = \{v_1, v_2\}$. However, $|S_{sub}| + |S_{cad}| = 0 + 2 < i$. So, node v_1 is not converting to state r yet. For $i = 2$, $S_{cad} = \{v_1, v_2\}$. Now, $|S_{sub}| + |S_{cad}| = 0 + 2 \geq i$. So, the algorithm returns r for node v_1. By symmetry, the output is the same for v_5.

Next, we consider node v_2. It is clear that $v_1, v_2, v_3, v_4 \in \Gamma(v_2)$. In addition, $\Gamma(v_2; v_3) = \Gamma(v_2)$; so, $v_2, v_3 \in S_{eql}$. Then, let us examine whether $v_1 \in S_{sub}$. Since $\Gamma(v_2; v_1) = \Gamma(v_1)$, we know that $\mathcal{F}_G^*(v_1, \Gamma(v_2; v_1)) = \mathcal{F}_G^*(v_1, \Gamma(v_1)) = r$. So, $v_1 \in S_{sub}$. Similarly, $v_4 \in S_{sub}$. In the Counting Phase, for $i = |S_{sub}| + |S_{eql}| = 4$, $S_{cad} = \{v_2\}$. However, $|S_{sub}| + |S_{cad}| = 2 + 1 < i$. So, node v_2 is not converting to state r yet. For $i = 3$, $S_{cad} = \{v_2\}$. Now, $|S_{sub}| + |S_{cad}| = 2 + 1 \geq i$. So, the algorithm returns r for node v_2. By symmetry, the output is the same for v_4. Last, for node v_3, clearly, $v_1, v_2, v_3, v_4, v_5 \in \Gamma(v_3)$ and $v_3 \in S_{eql}$. Since $\Gamma(v_3; v_1) = \Gamma(v_1)$, we know that

$$\mathcal{F}_G^*(v_1, \Gamma(v_3; v_1)) = \mathcal{F}_G^*(v_1, \Gamma(v_1)) = r.$$

So, $v_1 \in S_{sub}$. Similarly, $v_2, v_4, v_5 \in S_{sub}$. In the Counting Phase, for $i = |S_{sub}| + |S_{eql}| = 5$, $S_{cad} = \{v_3\}$. In this round, $|S_{sub}| + |S_{cad}| = 4 + 1 \geq i$. So, the algorithm returns r for node v_3. As we see, the algorithm will return r for all five nodes, and their thresholds are met.

Next, we need the following notations in proving the optimality of $\mathcal{F}_G^*$. To differentiate the node sets in $\mathcal{F}_G^*$ when the algorithm takes different nodes and views as inputs, denote them by $S_{sub}(v, \Gamma(v))$, $S_{eql}(v, \Gamma(v))$, $S_{max}(v, \Gamma(v))$ when taking node v and view $\Gamma(v)$ as inputs, respectively. Denote the *critical value* $\Phi_{\mathcal{F}_G^*}(v, \Gamma(v))$ as follows.

$$\Phi_{\mathcal{F}_G^*}(v, \Gamma(v)) = |S_{sub}(v, \Gamma(v))| + |S_{max}(v, \Gamma(v))|.$$

The critical value is the maximum permissible value of node v's threshold, such that its conversion to r is safe according to the algorithm. Obviously, according to the execution of $\mathcal{F}_G^*$, any two nodes u and v whose views $\Gamma(u)$ and $\Gamma(v)$ are equivalent, must have the same critical value, i.e., $\Phi_{\mathcal{F}_G^*}(v, \Gamma(v)) = \Phi_{\mathcal{F}_G^*}(u, \Gamma(u))$.

In the execution of Algorithm 2, while all nodes in S_{sub} are promised to convert to r in the Prediction Phase, for each i, in the Counting Phase, the algorithm verifies whether it is safe to convert those nodes in S_{eql} whose threshold is at most i to r. If it is safe, together with the inequalities in lines 12 and 14, it implies that it is safe for node v to convert to r. [2]

The following lemmas establish properties of the critical value, which will be useful in proving the main theorem of this section, Theorem 5.1.

LEMMA 5.1. *Given node v and its view $\Gamma(v)$, if $\Phi(v) \leq \Phi_{\mathcal{F}_G^*}(v, \Gamma(v))$, then $\mathcal{F}_G^*(v, \Gamma(v)) = r$; otherwise, $\mathcal{F}_G^*(v, \Gamma(v)) = s$.*

PROOF. Suppose the execution of the loop in the Counting Phase stops at $i = j$. Note that the loop executes in decreasing order of i, we have $j + 1 > |S_{sub}(v, \Gamma(v))| + |S_{cad}(j, v, \Gamma(v))| = j$. Since $S_{cad}(j, v, \Gamma(v))$ is assigned to $S_{max}(v, \Gamma(v))$, we have $\Phi_{\mathcal{F}_G^*}(v, \Gamma(v)) =$

[2] We note that an alternative version of the Counting Phase is that first check whether $|S_{sub}| + |S_{eql}| \geq \Phi(v)$, if it is true, then decrease the value of i from $|S_{sub}| + |S_{eql}|$ till $\Phi(v)$. This way, the sets S_{cad} and S_{max} may not be well-defined in some cases, which will create extra complexity in proving the monotonicity property.

$|S_{sub}(v,\Gamma(v))| + |S_{max}(j,v,\Gamma(v))| = j$. Thus, the algorithm returns r if and only if $\Phi_{\mathcal{F}^*_G}(v,\Gamma(v)) \geq \Phi(v)$. □

The following lemma states that the critical value $\Phi_{\mathcal{F}^*_G}(v,\Gamma(v))$ is non-decreasing in view $\Gamma(v)$. The intuition is that if a node has a larger view, it can predict more k-hop neighbors that convert to state r, and so it should have a higher critical value. However, based on the formula definition of $\Phi_{\mathcal{F}^*_G}(v,\Gamma(v))$, it consists of two parts where the first part $|S_{sub}(v,\Gamma(v))|$ is not monotone in view $\Gamma(v)$. We must carefully analyze the change between S_{sub} and S_{max} caused by the larger view.

LEMMA 5.2. *If* $\Gamma'(v) \subseteq \Gamma(v)$, *then* $\Phi_{\mathcal{F}^*_G}(v,\Gamma'(v)) \leq \Phi_{\mathcal{F}^*_G}(v,\Gamma(v))$.

PROOF. We prove it by induction. For node v, let $H_v(n,m)$ be the set of v's views $\Gamma(v)$ that has no more than n nodes and m edges. Note that any view in $H_v(n,m)$ is a connected simple graph. So, for any view $\Gamma(v) \in H_v(n,m)$, it has at least $n-1$ nodes and at most $\frac{(n-1)n}{2}$ edges. It is easy to verify that for views $\Gamma'(v) \subseteq \Gamma(v) \in H_v(2,1)$, the lemma holds. Assume that for $n \geq 2$ and m, for any views $\Gamma'(v) \subseteq \Gamma(v) \in H_v(n,m)$, the lemma holds. We will show that the lemma holds for two possible cases:

(1) any views $\Gamma'(v) \subseteq \Gamma(v) \in H_v(n+1,n)$, if $m = \frac{n(n-1)}{2}$,
(2) any views $\Gamma'(v) \subseteq \Gamma(v) \in H_v(n,m+1)$, if $m < \frac{n(n-1)}{2}$.

That is, if the view set $H_v(n,m)$ in the induction step contains views that are complete graphs with n nodes, then in Case 1 we show that the lemma holds when the number of nodes is increased by one, and the view graphs are trees. If the view set $H_v(n,m)$ does not contain views that are complete graphs, then in Case 2 we show that the lemma holds when the number of edges in the view graphs is increased by one. In both cases, we show that the nodes in $S_{sub}(v,\Gamma'(v))$ and $S_{max}(v,\Gamma'(v))$ will be in $S_{sub}(v,\Gamma(v))$ and $S_{max}(v,\Gamma(v))$. Therefore, $|S_{sub}(v,\Gamma'(v))| + |S_{max}(v,\Gamma'(v))| \leq |S_{sub}(v,\Gamma(v))| + |S_{max}(v,\Gamma(v))|$.

Case 1. To prove this case, it is sufficient to consider views $\Gamma'(v) \subsetneq \Gamma(v)$ such that view $\Gamma(v)$ is a tree of size $n+1$ and view $\Gamma'(v)$ is obtained by removing a degree-one node in $\Gamma(v)$ and the edge incident on the node. Other cases are covered in the induction step.

We partition $S_{max}(v,\Gamma'(v))$ into two node sets A and B. Let A contain the nodes in $S_{max}(v,\Gamma'(v))$ such that their corresponding nodes in $V(v)$ have the equivalent view as node v's view $\Gamma(v)$. So, $A \subseteq S_{eql}(v,\Gamma(v))$. Let $B = S_{max}(v,\Gamma'(v)) \setminus A$. We will show that $S_{sub}(v,\Gamma'(v)) \cup B \subseteq S_{sub}(v,\Gamma(v))$ and $A \subseteq S_{max}(v,\Gamma(v))$.

If $B = \emptyset$, then it is trivial that $B \subseteq S_{sub}(v,\Gamma(v))$. If $B \neq \emptyset$, for each node $u \in B$, since $S_{max}(v,\Gamma'(v)) \subseteq S_{eql}(v,\Gamma'(v))$, we know that $u \in S_{eql}(v,\Gamma'(v))$. Therefore, $\Gamma'(v;u) = \Gamma'(v)$. Except the return value, the execution of algorithm $\mathcal{F}^*_G$ is identical when it takes $(v,\Gamma'(v))$ and $(u,\Gamma'(v;u))$ as inputs. Then we have $u \in S_{max}(v,\Gamma'(v)) = S_{max}(u,\Gamma'(v;u))$, it implies Counting Phase of algorithm with input $(u,\Gamma'(v;u))$ stops at $i \geq \Phi(u)$. Hence $\mathcal{F}^*_G(u,\Gamma'(v;u)) = r$. The view of u's corresponding node in $V(v)$ is a sub-view of $\Gamma(v)$. Therefore, the number of nodes in $\Gamma(v;u)$ must be at most n. That is, $\Gamma(v;u) \in H_v(n,n-1)$. According to the induction step, it holds that $\Phi_{\mathcal{F}^*_G}(v,\Gamma'(v;u)) \leq \Phi_{\mathcal{F}^*_G}(v,\Gamma(v;u))$. Since $\Gamma'(v;u) \subseteq \Gamma(v;u)$, by Lemma 5.1, we have that $\mathcal{F}^*_G(u,\Gamma(v;u)) = r$, which indicates that $B \subseteq S_{sub}(v,\Gamma(v))$.

For each node $u \in S_{sub}(v,\Gamma'(v))$, we know that $\Gamma'(v;u) \subsetneq \Gamma'(v)$. As both views are trees, it implies that $\Gamma(v;u) \subsetneq \Gamma(v)$. Since $\Gamma(v;u) \in H(n,n-1)$, again, by the induction step and Lemma 5.1, we obtain that $\mathcal{F}^*_G(u,\Gamma(v;u)) = r$. So, $u \in S_{sub}(v,\Gamma(v))$. Therefore, we conclude that $S_{sub}(v,\Gamma'(v)) \cup B \subseteq S_{sub}(v,\Gamma(v))$.

Denote $S_{cad}(j,v,\Gamma(v))$ the set S_{cad} in the *For* loop of the Counting Phase when $i = j$. In the Counting Phase of $\mathcal{F}^*_G(v,\Gamma(v))$, if the inequality in Line 12 holds at a *For* loop $i > \Phi(A)$, then the set $S_{max}(v,\Gamma(v))$ contains all nodes that belong to $S_{eql}(v,\Gamma(v))$ and their thresholds are at most i. Therefore, $A \subseteq S_{max}(v,\Gamma(v))$. Otherwise, we will show that the Counting Phase will stop at the loop $i = \Phi(A)$ and $A \subseteq S_{max}(v,\Gamma(v))$ as well. When $i = \Phi(A)$, set $S_{cad}(i,v,\Gamma(v))$ contains all nodes that belong to $S_{eql}(v,\Gamma(v))$ and their thresholds are at most $\Phi(A)$. Therefore, $A \subseteq S_{cad}(i,v,\Gamma(v))$. Hence,

$$\begin{aligned}
&|S_{sub}(v,\Gamma(v))| + |S_{cad}(i,v,\Gamma(v))| \\
\geq&(|S_{sub}(v,\Gamma'(v))| + |B|) + |A| \\
=&|S_{sub}(v,\Gamma'(v))| + |S_{max}(v,\Gamma'(v))| \\
\geq&\Phi(S_{max}(v,\Gamma'(v))) \geq \Phi(A) = i.
\end{aligned}$$

Therefore, the inequality in Line 12 holds, and the set $S_{cad}(i,v,\Gamma(v))$ is assigned to $S_{max}(v,\Gamma(v))$. Hence, $A \subseteq S_{max}(v,\Gamma(v))$. In conclusion,

$$\begin{aligned}
\Phi_{\mathcal{F}^*_G}(v,\Gamma'(v)) &= |S_{sub}(v,\Gamma'(v))| + |S_{max}(v,\Gamma'(v))| \\
&= (|S_{sub}(v,\Gamma'(v))| + |B|) + |A| \\
&\leq |S_{sub}(v,\Gamma(v))| + |S_{max}(v,\Gamma(v))| \\
&= \Phi_{\mathcal{F}^*_G}(v,\Gamma(v)).
\end{aligned}$$

Case 2. To prove this case, it is sufficient to consider views $\Gamma'(v) \subsetneq \Gamma(v)$ such that view $\Gamma(v)$ is a graph with n nodes and $m+1$ edges, and view $\Gamma'(v)$ is obtained by removing an edge in $\Gamma(v)$. Other cases are covered in the induction step. Distinct from Case 1, in this case, there may exist a node in $S_{sub}(v,\Gamma'(v))$ and its corresponding node is in $S_{eql}(v,\Gamma(v))$.

First, we show that $S_{max}(v,\Gamma'(v)) \subseteq S_{sub}(v,\Gamma(v)) \cup S_{eql}(v,\Gamma(v))$. For any $u \in S_{max}(v,\Gamma'(v)) \subseteq S_{eql}(v,\Gamma'(v))$, we have that $\Gamma'(v;u) = \Gamma'(v)$. Similar to Case 1, the execution of the algorithm is identical except the return value when it takes $(v,\Gamma'(v))$ and $(u,\Gamma'(v;u))$ as inputs. The loop in Counting Phase stops at some i which is larger than $\Phi(S_{max}(v,\Gamma'(v)))$ and it is larger than $\Phi(u)$. By Algorithm 2, we have $\mathcal{F}^*_G(u,\Gamma'(v;u)) = r$. If $\Gamma(v;u) = \Gamma(v)$, we have that $u \in S_{eql}(v,\Gamma(v))$. Otherwise, $\Gamma(v;u) \subsetneq \Gamma(v)$, together with $\Gamma'(v;u) \subseteq \Gamma(v;u)$, $\Gamma'(v;u) = \Gamma'(v) \subseteq \Gamma(v)$, and that $\Gamma'(v)$ is obtained by removing an edge in $\Gamma(v)$, we have that $\Gamma'(v;u) = \Gamma(v;u)$. It implies that $\mathcal{F}^*_G(u,\Gamma(v;u)) = \mathcal{F}^*_G(u,\Gamma'(v;u)) = r$. Therefore, $u \in S_{sub}(v,\Gamma(v))$.

Second, we show that $S_{sub}(v,\Gamma'(v)) \subseteq S_{sub}(v,\Gamma(v)) \cup S_{eql}(v,\Gamma(v))$. For any $u \in S_{sub}(v,\Gamma'(v))$, we know that $\mathcal{F}^*_G(u,\Gamma'(v;u)) = r$. If $\Gamma(v;u) = \Gamma(v)$, we have that $u \in S_{eql}(v,\Gamma(v))$. Otherwise, $\Gamma(v;u) \subsetneq \Gamma(v)$. Since $\Gamma'(v;u) \subseteq \Gamma(v;u) \in H_v(n,m)$, according to the induction step and Lemma 5.1, we have that $\mathcal{F}^*_G(u,\Gamma(v;u)) = r$. Therefore, $u \in S_{sub}(v,\Gamma(v))$.

In conclusion, we have shown that

$$S_{sub}(v,\Gamma'(v)) \cup S_{max}(v,\Gamma'(v)) \subseteq S_{sub}(v,\Gamma(v)) \cup S_{eql}(v,\Gamma(v)). \tag{4}$$

Next, we will show that for any node $u \in S_{sub}(v,\Gamma'(v)) \cup S_{max}(v,\Gamma'(v))$ such that its corresponding node $u \in S_{eql}(v,\Gamma(v))$, it must be the case that $u \in S_{max}(v,\Gamma(v))$.

Denote $\bar{w}_s$ and $\bar{w}_m$ the nodes that have the largest thresholds in sets $S_{sub}(v,\Gamma'(v))$ and $S_{max}(v,\Gamma'(v))$, respectively. We consider two sub-cases.

Case 2(*i*). $\Phi(\bar{w}_m) \geq \Phi(\bar{w}_s)$. On the one hand, we know that for any node $u \in S_{sub}(v,\Gamma'(v)) \cup S_{max}(v,\Gamma'(v))$, $\Phi(u) \leq \Phi(\bar{w}_m)$. On the other hand, when the algorithm takes $(v,\Gamma(v))$ as input, it will stop the latest at the *For* loop $i = \Phi(\bar{w}_m)$, because that

$$\begin{aligned}
&|S_{sub}(v,\Gamma(v))| + |S_{cad}(\Phi(\bar{w}_m), v, \Gamma(v))| \\
=&|S_{sub}(v,\Gamma(v))| + |\{z | z \in S_{eql}(v,\Gamma(v)), \Phi(z) \leq \Phi(\bar{w}_m)\}| \\
\geq&|S_{sub}(v,\Gamma(v))| + |\{z | z \in S_{eql}(v,\Gamma(v)), \Phi(z) \leq \Phi(\bar{w}_m)\} \\
&\quad \cap (S_{sub}(v,\Gamma'(v)) \cup S_{max}(v,\Gamma'(v)))| \\
\geq&|S_{sub}(v,\Gamma'(v))| + |S_{max}(v,\Gamma'(v))| \qquad \text{(By (4))} \\
\geq&\Phi(\bar{w}_m).
\end{aligned}$$

Therefore, $S_{cad}(\Phi(\bar{w}_m), v, \Gamma(v))$ is assigned to $S_{max}(v,\Gamma(v))$ and $u \in S_{max}(v,\Gamma(v))$.

Case 2(*ii*). $\Phi(\bar{w}_m) < \Phi(\bar{w}_s)$. Since $\bar{w}_s \in S_{sub}(v,\Gamma'(v))$, we know that $\mathcal{F}^*_G(\bar{w}_s, \Gamma'(v;\bar{w}_s)) = r$. It implies $\bar{w}_s \in S_{max}(\bar{w}_s, \Gamma'(v;\bar{w}_s))$. According to the algorithm, we have

$$|S_{sub}(\bar{w}_s, \Gamma'(v;\bar{w}_s))| + |S_{max}(\bar{w}_s, \Gamma'(v;\bar{w}_s))| \geq \Phi(\bar{w}_s) \tag{5}$$

For $u \in S_{sub}(\bar{w}_s, \Gamma'(v;\bar{w}_s)) \cup S_{max}(\bar{w}_s, \Gamma'(v;\bar{w}_s))$, we always have $\mathcal{F}^*_G(u, \Gamma'(v,\bar{w}_s,u)) = r$. Since $\Gamma'(v,\bar{w}_s,u) \subseteq \Gamma'(v;u) \in H_v(n,m)$, according to the induction step and Lemma 5.1, we have that $\mathcal{F}^*_G(u,\Gamma'(v;u)) = r$. If $\Gamma'(v;u) = \Gamma'(v)$, we have $u \in S_{max}(v,\Gamma'(v))$. Otherwise, we have $u \in S_{sub}(v,\Gamma'(v))$. Thus, we have proved

$$\begin{aligned}
&S_{sub}(\bar{w}_s, \Gamma'(v;\bar{w}_s)) \cup S_{max}(\bar{w}_s, \Gamma'(v;\bar{w}_s)) \\
\subseteq&S_{sub}(v,\Gamma'(v)) \cup S_{max}(v,\Gamma'(v))
\end{aligned}$$

We plug it into Formula (5) and get

$$|S_{sub}(v,\Gamma'(v))| + |S_{max}(v,\Gamma'(v))| \geq \Phi(\bar{w}_s).$$

Similar to Case 1, it suffices to show the inequality in Line 12 holds at $i = \Phi(\bar{w}_s)$. According to Formula (4) and the definition of $\bar{w}_s$, we have

$$\begin{aligned}
&S_{sub}(v,\Gamma'(v)) \cup S_{max}(v,\Gamma'(v)) \\
\subseteq&S_{sub}(v,\Gamma(v)) \cup S_{cad}(\Phi(\bar{w}_s), v, \Gamma(v))
\end{aligned} \tag{6}$$

Thus, we have $|S_{sub}(v,\Gamma(v))| + |S_{cad}(\Phi(\bar{w}_s), v, \Gamma(v))| \geq \Phi(\bar{w}_s) = \Phi(S_{cad}(\Phi(\bar{w}_s, v, \Gamma(v))))$. The inequality in Line (12) holds. It implies $S_{cad}(\Phi(\bar{w}_s, v, \Gamma(v))) \subseteq S_{max}(v,\Gamma(v))$. Plug it into Formula (6), we get $\Phi_{\mathcal{F}^*_G}(v,\Gamma'(v)) \leq \Phi_{\mathcal{F}^*_G}(v,\Gamma(v))$. □

LEMMA 5.3. *$\mathcal{F}^*_G$ is monotone.*

PROOF. Consider any view $\Gamma(v)$ and its sub-view $\Gamma'(v)$. By Lemma 5.2, we get that $\Phi_{\mathcal{F}^*_G}(v,\Gamma'(v)) \leq \Phi_{\mathcal{F}^*_G}(v,\Gamma(v))$. If $\mathcal{F}^*_G(v,\Gamma'(v)) = r$, then it must be that $\Phi(v) \leq \Phi_{\mathcal{F}^*_G}(v,\Gamma'(v))$. Hence, $\Phi(v) \leq \Phi_{\mathcal{F}^*_G}(v,\Gamma(v))$. So, when the algorithm takes $\Gamma(v)$ as input, we obtain that $\mathcal{F}^*_G(v,\Gamma(v)) = r$. Therefore, $\mathcal{F}^*_G$ is monotone. □

Finally, we are ready to prove the second main result.

THEOREM 5.1. *$\mathcal{F}^*_G$ is optimal in the General Threshold model.*

PROOF. Firstly, we show that $\mathcal{F}^*_G$ is safe. For any graph, suppose that $\mathcal{F}^*_G(v,\Gamma(v)) = r$. According to the execution of the algorithm, for any node $u \in S_{sub}(v,\Gamma(v)) \cup S_{max}(v,\Gamma(v))$, we have that $\mathcal{F}^*_G(u,\Gamma(v;u)) = r$. Since $\mathcal{F}^*_G$ is monotone, we have that $\mathcal{F}^*_G(u,\Gamma(u)) = r$. So, each node in $S_{sub}(v,\Gamma(v)) \cup S_{max}(v,\Gamma(v))$ will convert to state r. Therefore, $\mathcal{F}^*_G$ is safe.

Secondly, we prove that $\mathcal{F}^*_G$ is optimal by contradiction. We will show that for any safe algorithm $\mathcal{F}$ and any view $\Gamma(v)$, if $\mathcal{F}^*_G(v,\Gamma(v)) = s$, then $\mathcal{F}(v,\Gamma(v)) = s$. Suppose that there exists a safe algorithm $\mathcal{F}$ and a view $\Gamma(v)$ such that $\mathcal{F}(v,\Gamma(v)) = r$ and $\mathcal{F}^*_G(v,\Gamma(v)) = s$. Amongst these views $\Gamma(v)$, we consider the ones that have the least number of nodes; amongst these views, we take the one that has the least number of edges and denote it as $\Gamma'(v)$ (break ties arbitrarily). Consider applying algorithm $\mathcal{F}$ to a graph G that has the same topology as $\Gamma'(v)$. Define

- S_r : the set of nodes in G that convert to state r under $\mathcal{F}$;
- $S_{nsub} : V(v) \setminus (S_{eql}(v,\Gamma(v)) \cup S_{sub}(v,\Gamma(v)))$.

According to the least number of nodes and edges assumption of $\Gamma'(v)$, for each $u \in S_{nsub}$, it holds that $\mathcal{F}(u,\Gamma'(v;u)) = s$. So, $S_{nsub} \cap S_r = \emptyset$ and further that $S_r \subseteq S_{sub}(v,\Gamma'(v)) \cup S_{eql}(v,\Gamma'(v))$. As $\mathcal{F}^*_G(v,\Gamma'(v)) = s$, we can get that for any $S \subseteq S_{eql}(v,\Gamma'(v))$ and $\Phi(S) \geq \Phi(v)$ it holds that $\Phi(S) > |S_{sub}(v,\Gamma'(v))| + |S|$. We set $S = S_r \cap S_{eql}(v,\Gamma'(v))$. Since $v \in S_r$ and $v \in S_{eql}(v,\Gamma'(v))$, we have $\Phi(S) \geq \Phi(v)$. As a result,

$$\begin{aligned}
&\Phi(S_r \cap S_{eql}(v,\Gamma'(v))) \\
>&|S_{sub}(v,\Gamma'(v))| + |S_r \cap S_{eql}(v,\Gamma'(v))| \\
\geq&|S_r \cap S_{sub}(v,\Gamma'(v))| + |S_r \cap S_{eql}(v,\Gamma'(v))| = |S_r|.
\end{aligned}$$

It indicates that at least one node in S_r converts to state r without enough neighbors in state r. Hence, $\mathcal{F}$ is not safe and we get a contradiction. □

We define E_{max} and N_{max} as the maximum number of edges and nodes in a node's view. In the Prediction Phase in Algorithm 2, the number of edges in a sub-view decreases by at least one. Then the number of levels of program recursive calls is at most E_{max}. Therefore, Line 3 is run at most $E_{max}!$ times. In addition, note that each Counting Phase needs $O(N^2_{max})$ time. Consequently, the time complexity of Algorithm 2 is $O(E_{max}! \cdot N^2_{max})$.

We have assumed that all nodes are set to s initially. Actually, our protocol can be modified to handle the general case that some nodes start with state r. If there are nodes that start with state r, then we can consider that the threshold of these nodes is 0. That is, no matter what their neighbors do, they would always commit to the state r. Plus, this fact is known to other nodes. When a node runs Algorithm 2, the nodes start with state r would belong to either S_{sub} or S_{eql} in the Prediction Phase.

6 CONCLUSION AND FUTURE WORK

In this paper, we considered the optimization problem that maximizing node conversion in Influence Networks under the premise that their conversion is safe. We generalized the literature in two-fold: from 1-hop to k-hop neighborhood information and from uniform threshold to general thresholds. We designed optimal local decision algorithms in both models, and the proof of their optimality provides insights into the nodes' recursive reasoning about their neighbors' actions. The monotonicity properties of these algorithms may find useful in future works.

Several directions remain open. First, due to the nature of the problem, the optimal algorithms are exponential time on general graphs. It remains open whether poly-time algorithms exist for simpler graph structures. We conjecture that the problem remains intractable on tree graphs but is polynomial-time solvable on paths. Second, this paper deals with the asynchronous setting in which each node has its independent clock so that its conversion is a one-shot process. In case the nodes are synchronized with a global clock, the network evolves over a series of rounds. Then, with the nodes' historical states, the optimal algorithms can infer additional information according to their conversion at different rounds and eventually enable more nodes to convert to state r. Moreover, the extent to which algorithms are having access to the last time epochs will likely have an impact on the conversion rate of the optimal algorithms.

7 ACKNOWLEDGMENTS

Jie Zhang was partially supported by a Leverhulme Trust Research Project Grant (2021 – 2024) and an EPSRC grant (EP/W014912/1). Zihe Wang was partially supported by the National Natural Science Foundation of China (Grant No. 62172422); Beijing Outstanding Young Scientist Program (No. BJJWZYJH012019100020098); Intelligent Social Governance Platform, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China. We also thank the anonymous reviewers for their valuable suggestions.

REFERENCES

[1] Joseph B Bak-Coleman, Mark Alfano, Wolfram Barfuss, Carl T Bergstrom, Miguel A Centeno, Iain D Couzin, Jonathan F Donges, Mirta Galesic, Andrew S Gersick, Jennifer Jacquet, et al. 2021. Stewardship of global collective behavior. *Proceedings of the National Academy of Sciences* 118, 27 (2021), e2025764118.

[2] Renaud Bastien and Pawel Romanczuk. 2020. A model of collective behavior based purely on vision. *Science advances* 6, 6 (2020), eaay0792.

[3] Sébastien Bubeck and Thomas Budzinski. 2020. Coordination without communication: optimal regret in two players multi-armed bandits. In *Conference on Learning Theory*. PMLR, 916–939.

[4] Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. 2012. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial informatics* 9, 1 (2012), 427–438.

[5] Michael Suk-Young Chwe. 1999. Structure and strategy in collective action 1. *American journal of sociology* 105, 1 (1999), 128–156.

[6] Michael Suk-Young Chwe. 2000. Communication and coordination in social networks. *The Review of Economic Studies* 67, 1 (2000), 1–16.

[7] Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. 2019. Tarmac: Targeted multi-agent communication. In *International Conference on Machine Learning*. PMLR, 1538–1546.

[8] Jordi Delgado. 2002. Emergence of social conventions in complex networks. *Artificial intelligence* 141, 1-2 (2002), 171–185.

[9] Danny Dolev and H. Raymond Strong. 1983. Authenticated algorithms for Byzantine agreement. *SIAM J. Comput.* 12, 4 (1983), 656–666.

[10] Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. 1988. Consensus in the presence of partial synchrony. *Journal of the ACM (JACM)* 35, 2 (1988), 288–323.

[11] David Easley and Jon Kleinberg. 2010. *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press.

[12] Sergey Gavrilets and Mahendra Duwal Shrestha. 2020. Evolving institutions for collective action by selective imitation and self-interested design. *Evolution and Human Behavior* (2020).

[13] Michael R Genesereth, Matthew L Ginsberg, and Jeffrey S Rosenschein. 1988. Cooperation without communication. In *Readings in distributed artificial Intelligence*. Elsevier, 220–226.

[14] Irene Giardina. 2008. Collective behavior in animal groups: theoretical models and empirical studies. *HFSP journal* 2, 4 (2008), 205–219.

[15] Parameswaran Gopikrishnan, Bernd Rosenow, Vasiliki Plerou, and H Eugene Stanley. 2001. Quantifying and interpreting collective behavior in financial markets. *Physical Review E* 64, 3 (2001), 035106.

[16] Valentin Goranko, Antti Kuusisto, and Raine Rönnholm. 2020. Rational coordination with no communication or conventions. *Journal of Logic and Computation* 30, 6 (2020), 1183–1211.

[17] Paul Harrenstein, Wiebe van der Hoek, John-Jules Meyer, and Cees Witteveen. 2001. Boolean games. In *Proceedings of the 8th conference on Theoretical aspects of rationality and knowledge*. 287–298.

[18] Shuyue Hu and Ho-fung Leung. 2017. Achieving Coordination in Multi-Agent Systems by Stable Local Conventions under Community Networks.. In *IJCAI*. 4731–4737.

[19] Matthew O Jackson and Yves Zenou. 2015. Games on networks. In *Handbook of game theory with economic applications*. Vol. 4. Elsevier, 95–163.

[20] Paan Jindapon and Zhe Yang. 2020. Free riders and the optimal prize in public-good funding lotteries. *Journal of Public Economic Theory* 22, 5 (2020), 1289–1312.

[21] Leslie Lamport, Robert Shostak, and Marshall Pease. 2019. The Byzantine generals problem. In *Concurrency: the works of leslie lamport*. 203–226.

[22] Vadim Levit, Zohar Komarovsky, Tal Grinshpoun, and Amnon Meisels. 2018. Incentive-based search for efficient equilibria of the public goods game. *Artificial Intelligence* 262 (2018), 142–162.

[23] Yuezhou Lv and Thomas Moscibroda. 2015. Incentive Networks. In *AAAI,*, Blai Bonet and Sven Koenig (Eds.). 1270–1276.

[24] Yuezhou Lv and Thomas Moscibroda. 2015. Local information in influence networks. In *International Symposium on Distributed Computing*. 292–308.

[25] M. Macy. 1991. Chains of Cooperation: Threshold Effects in Collective Action. *American Sociological Review* 56, 6 (1991), 730–744.

[26] David Mateo, Nikolaj Horsevad, Vahid Hassani, Mohammadreza Chamanbaz, and Roland Bouffanais. 2019. Optimal network topology for responsive collective behavior. *Science advances* 5, 4 (2019), eaau0999.

[27] Stephen Morris. 2000. Contagion. *Review of Economic Studies* 67, 1 (2000), 57–78.

[28] Stephen Morris and Hyun Song Shin. 2002. Social value of public information. *american economic review* 92, 5 (2002), 1521–1534.

[29] Kene Boun My, Camille Cornand, and Rodolphe Dos Santos Ferreira. 2021. Public information and the concern for coordination. *Journal of Behavioral and Experimental Economics* 93 (2021), 101710.

[30] Elinor Ostrom. 2014. Collective action and the evolution of social norms. *Journal of Natural Resources Policy Research* 6, 4 (2014), 235–252.

[31] PC Sachin and Dua Amit. 2022. A Survey on Byzantine Agreement Algorithms in Distributed Systems. In *Intelligent Systems and Sustainable Computing*. Springer, 495–506.

[32] Kazunori Sakurama, Shun-Ichi Azuma, and Toshiharu Sugie. 2018. Multiagent coordination via distributed pattern matching. *IEEE Trans. Automat. Control* 64, 8 (2018), 3210–3225.

[33] Fernando P Santos, Samuel Francisco Mascarenhas, Francisco C Santos, Filipa Correia, Samuel Gomes, and Ana Paiva. 2019. Outcome-based Partner Selection in Collective Risk Dilemmas.. In *AAMAS*. 1556–1564.

[34] Paul Schermerhorn and Matthias Scheutz. 2006. Social coordination without communication in multi-agent territory exploration tasks. In *AAMAS*. 654–661.

[35] Leigh Tesfatsion and Kenneth L Judd. 2006. *Handbook of computational economics: agent-based computational economics*. Vol. 2 (829–1660). Elsevier.

[36] Stanley Wasserman. 1994. *Social Network Analysis: Methods and Applications*. Cambridge University Press.

[37] Barry Brian Werger. 1999. Cooperation without deliberation: A minimal behavior-based approach to multi-robot teams. *Artificial Intelligence* 110, 2 (1999), 293–320.

[38] Barry Brian Werger and Maja J Matarić. 2001. From insect to internet: Situated control for networked robot teams. *Annals of Mathematics and Artificial Intelligence* 31, 1 (2001), 173–197.

[39] Michael Wooldridge, Ulle Endriss, Sarit Kraus, and Jérôme Lang. 2013. Incentive engineering for Boolean games. *Artificial Intelligence* 195 (2013), 418–439.

[40] Lan Yang, Zhiwu Li, and Alessandro Giua. 2019. Influence minimization in linear threshold networks. *Automatica* 100 (2019), 10–16.

[41] Ercan Yildiz, Asuman Ozdaglar, Daron Acemoglu, Amin Saberi, and Anna Scaglione. 2013. Binary opinion dynamics with stubborn agents. *ACM Transactions on Economics and Computation (TEAC)* 1, 4 (2013), 1–30.