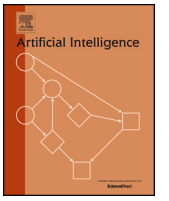
Artificial Intelligence 341 (2025) 104297

Contents lists available at ScienceDirect

Artificial Intelligence

journal homepage: www.elsevier.com/locate/artint



On the computation of mixed strategies for security games with general defending requirements

Rufan Bai [a], Haoxing Lin [b], Xiaowei Wu [c,*], Minming Li [d], Weijia Jia [e]

[a] Southeast University, Nanjing, China
[b] National University of Singapore, Singapore
[c] University of Macau, Macau, China
[d] City University of Hong Kong, Hong Kong, China
[e] BNU-UIC Institute & Beijing Normal University (Zhuhai), Zhuhai, China

ARTICLE INFO

Keywords:
Security games
Mixed strategy
Efficient algorithm
Compact representation
Resource allocation

ABSTRACT

The Stackelberg security game is played between a defender and an attacker, where the defender needs to allocate a limited amount of resources to multiple targets in order to minimize the loss due to adversarial attacks by the attacker. While allowing targets to have different values, classic settings often assume uniform requirements for defending the targets. This enables existing results that study mixed strategies (randomized allocation algorithms) to adopt a *compact representation* of the mixed strategies.

In this work, we initiate the study of mixed strategies for security games in which the targets can have different defending requirements. In contrast to the case of uniform defending requirements, for which an optimal mixed strategy can be computed efficiently, we show that computing the optimal mixed strategy is NP-hard for the general defending requirements setting. However, we show strong upper and lower bounds for the optimal mixed strategy defending result. Additionally, we extend our analysis to study uniform attack settings on these security games.

We propose an efficient close-to-optimal Patching algorithm that computes mixed strategies using only a few pure strategies. Furthermore, we study the setting when the game is played on a network and resource sharing is enabled between neighboring targets. We show the effectiveness of our algorithm in various large real-world datasets, addressing both uniform and general defending requirements.

1. Introduction

Stackelberg security games have recently garnered significant interest from the game theory community and combinatorial optimization experts due to their extensive applications in real-world scenarios, such as patrolling [1,2], forest protection [3,4], defense coordination [5]. These classic security games typically model the interaction as a Stackelberg game [6,7], involving two players: the *defender* (leader) who commits to a defending strategy, and the *attacker* (follower), who observes and then reacts to this strategy. This paper concentrates on zero-sum games [8,9], where the objective is to defend multiple valuable targets. Each target u is associated with a value α_u, indicating the potential loss due to a successful attack on the target, and a threshold θ_u, representing the resources

* Corresponding author.
E-mail addresses: rfbai@seu.edu.cn (R. Bai), haoxing.lin@comp.nus.edu.sg (H. Lin), xiaoweiwu@um.edu.mo (X. Wu), minming.li@cityu.edu.hk (M. Li), jiawj@uic.edu.cn (W. Jia).

https://doi.org/10.1016/j.artint.2025.104297
Received 4 April 2024; Received in revised form 11 November 2024; Accepted 1 February 2025

Available online 10 February 2025

required to defend the target. A target u is considered secure if it receives resources at least θ_u, preventing any loss from attacks. Under this framework, the defender must strategize the allocation of limited resources across targets, while the attacker selects a target based on the defender's allocation. The defender's goal is to minimize the potential loss, which we define as the defending result of the allocation strategy.

The allocation strategies are often categorized into *pure strategies* and *mixed strategies*. An allocation is termed a pure strategy when it is deterministic, and a mixed strategy when it is randomized. Formally, a mixed strategy is defined as a probability distribution over a set of pure strategies. It is a common observation that mixed strategies yield defending results superior to those of the best pure strategy.

Example 1.1. Consider a scenario with targets $\{a, b, c, d\}$, each having a defending requirement (threshold) of 1. The values for targets $\{a, b, c\}$ are 3, and for target d, it is 1. With a total resource $R = 2$, every pure strategy leads to a defending result of 3, as there will always be one target among $\{a, b, c\}$ with insufficient defending resources. In contrast, a mixed strategy that applies each of the three pure strategies $(1, 1, 0, 0)$, $(1, 0, 1, 0)$, and $(0, 1, 1, 0)$ with probability $1/3$ achieves a defending result of 1.

Most existing works on mixed strategies in security games assume uniform thresholds [4,10–12], where all targets have the same defending requirements, i.e. $\theta_u = 1, \forall u \in V$. The example we show above has uniform thresholds. This allows for the representation of each mixed strategy through their corresponding *compact representation*, where the allocation to targets is no longer binary. Instead, a target receiving resources below its threshold is considered *fractionally* defended when evaluating the loss from an attack. Korzhyk et al. [12] demonstrated that any compact representation can be converted into a mixed strategy using $O(n^2)$ pure strategies that achieves the same defending result as the compact representation, where n is the number of targets. In this paper, we explore scenarios with non-uniform target thresholds, i.e., varying defending requirements among targets. Consider, for instance, the allocation of vaccines to cities (targets) in a virus defense scenario, where the defending requirement for each city depends on its population, leading to significant variance. This raises a critical question about the applicability of compact representation under these conditions:

Can every compact representation be transformed into a mixed strategy that achieves the same defending result when targets have general thresholds?

Regrettably, we show that this may not always be feasible.

1.1. Our contribution

We show that when targets have different defending requirements, the set of compact representations and mixed strategies are no longer equivalent. We henceforth refer to a compact representation as a *fractional strategy*. We formalize mixed and fractional strategies in security games with varied defending requirements, and explore their relationships and distinctions.

Mixed vs. fractional. A key contribution of our study is the theoretical exploration of the relation between mixed and fractional strategies. We denote the defending result of the optimal mixed strategy using total resource R as $\mathsf{OPT}_m(R)$, and that of the optimal fractional strategy as $\mathsf{OPT}_f(R)$. We establish that while computing $\mathsf{OPT}_m(R)$ is NP-hard, it holds that $\mathsf{OPT}_m(R) \geq \mathsf{OPT}_f(R)$. Given that $\mathsf{OPT}_f(R)$ can be determined through linear programming, it serves as a lower bound for $\mathsf{OPT}_m(R)$. Crucially, we demonstrate that for any given total resource R, a mixed strategy can be identified with a defending result no greater than $\mathsf{OPT}_f(R - \theta_{\max})$, where $\theta_{\max}$ represents the highest threshold among the targets. Furthermore, we introduce a polynomial-time algorithm capable of computing such a mixed strategy that uses $O(n^2)$ pure strategies. Interestingly, for uniform thresholds, the mixed strategy achieves a defending result $\mathsf{OPT}_f(R)$, i.e., our analysis re-produces the result of Korzhykcp et al. [12]. By proving the convexity of the function $\mathsf{OPT}_f(\cdot)$, we infer that when R significantly exceeds $\theta_{\max}$, the values of $\mathsf{OPT}_m(R)$ and $\mathsf{OPT}_f(R)$ closely converge, admitting a very small additive gap (see Section 3.3 for more details). Therefore, we establish the near-equivalence between mixed and fractional strategies in security games with general defending requirements.

Algorithm with small support. In practical applications, a mixed strategy employing a limited number of pure strategies is often preferable. For instance, deploying a mixed strategy that utilizes $\omega(n)$ pure strategies becomes impractical when the number of targets n is large. Therefore, our research focuses on devising mixed strategies with small *supports*. Inspired by the Double Oracle algorithm by Jain et al. [13] and the column generation techniques [14–16], we introduce the Patching algorithm. This algorithm iteratively identifies and incorporates new pure strategies to enhance the defense of poorly protected nodes within the current mixed strategy framework. We demonstrate that, for a given set of pure strategies D with bounded size, our algorithm computes an optimal mixed strategy with support D in polynomial time relative to $|D|$.

Resource sharing. Our investigation extends to scenarios involving resource sharing within a network, a concept inspired by applications in patrolling and surveillance camera deployment [17–19]. In such models, targets are depicted as network nodes, allowing for the sharing of a portion of resources allocated to neighboring nodes during an attack. While similar studies (e.g. [20]) have explored models with general thresholds focusing solely on pure strategies, our work delves into the implications of resource sharing. We reveal that the gap between $\mathsf{OPT}_m(R)$ and $\mathsf{OPT}_f(R)$ can become arbitrarily large with resource sharing, undermining the feasibility of approximating mixed strategies from rounding the fractional ones. Nonetheless, the Patching algorithm remains applicable, efficiently computing mixed strategies within this context. We further ascertain that, under specific conditions, the algorithm is capable of making progress toward reducing the defending result.

Table 1
Summary of existing models.

Model	Limited Budget		Threshold		Mixed Strategy		Resource Sharing		Attack Type	
	Yes	No	General	Uniform	Yes	No	Duplicated	Reallocate	Adversary	Uniform
Kiekintveld et al. [11]	✓			✓	✓				✓	
Korzhy et al. [12]	✓			✓	✓				✓	
Bai et al. [19]	✓		✓			✓		✓	✓	
Bai et al. [24]	✓		✓			✓		✓	✓	✓
An et al. [25]	✓			✓	✓				✓	
Vorobeychik et al. [17]	✓		✓		✓				✓	
Conitzer et al. [36]		✓	✓		✓				✓	
Aspnes et al. [21]		✓		✓		✓				✓
Kumar et al. [22]		✓	✓			✓				✓
Gan et al. [27]	✓		✓			✓	✓		✓	
Li et al. [20]	✓		✓			✓	✓		✓	
Yin et al. [18]	✓		✓			✓		✓	✓	
Ours (isolated)	✓		✓		✓				✓	✓
Ours (shared)	✓		✓		✓		✓		✓	✓

Uniform attack. We further extend our analysis to deal with uniform attacks, wherein the attacker attacks each target with equal probability. In this scenario, we show that the optimal mixed strategy is a pure strategy, and its computation (even without resource sharing), is NP-hard. Nonetheless, we identify a tight connection between the model and the knapsack problem, using which we establish lower and upper bounds for the defending results: $\mathsf{OPT}_f^{\text{uni}}(R) \le \mathsf{OPT}_m^{\text{uni}}(R) \le \mathsf{OPT}_f^{\text{uni}}(R - \theta_{\max})$.

Experiments. To validate our theoretical findings, we conducted comprehensive experiments using several large real-world datasets. The outcomes demonstrate the Patching algorithm's efficiency in computing mixed strategies with small support, such as utilizing only 5 pure strategies. These strategies significantly enhance the defending results beyond those achievable with optimal pure strategies and approach optimality in numerous instances.

The remainder of this paper is organized as follows: Section 2 provides a detailed model description. Sections 3.1 to 3.3 delve into the relation between mixed and fractional strategies, laying the groundwork for our theoretical contributions. Section 3.5 discusses some challenges encountered in the resource-sharing context. The Patching algorithm is detailed in Section 4. The results for uniform attacks are presented in Section 5, which are followed by a presentation of our experimental findings in Section 6.

1.2. Other related works

Motivated by applications aimed at halting the spread of viruses, network security games with contagious attacks have garnered considerable attention in recent years [21,22,9,23,19,24]. In these models, an attack on a node can propagate to its neighbors, with the loss evaluated across all affected nodes. Aspnes et al. [21] conceptualized the inoculation challenge as a network security game, analyzing the pure Nash equilibrium with a focus on uniform thresholds. Kumar et al. [22] extended this work to consider general thresholds but did not account for the common real-world constraint of limited resources. Several studies have addressed security games under a limited budget [19,24–26], emphasizing the finite nature of defending resources. Additionally, some research has explored security games with resource-sharing mechanisms [27,20]. Gan et al. [27] introduced a network security game where allocating resources to a target also benefits its neighbors. Li et al. [20] proposed a model where a node's defense capability is determined by both its allocated resources and a linear combination of resources from its neighbors. Kroupa et al. [28] studied the continuous games and proposed a type of multiple oracle algorithm. There are investigations into security games where resource sharing is dynamic, requiring time for neighboring nodes to share resources [29,18], and studies on multi-defender games, with each defender responsible for a single target [30–32]. There are also some studies on multiple defender Stackelberg security games that consider robust solutions [33], focus on resource allocation based on risk sharing [34], or tackle scheduling problems [35].

To facilitate a clear comparison with closely related models, we summarize the main features of these models in Table 1.

2. Preliminaries

In this section we present the model we study. We define our model in the most general form, i.e., including the network structure and with resource sharing, and consider the model without resource sharing as a restricted setting. We model the network as an undirected connected graph $G(V, E)$, where each node $u \in V$ has a *threshold* θ_u that represents the defending requirement, and a *value* α_u that represents the possible damage due to an attack at node u. Each edge $e \in E$ is associated with a weight w_{uv}, which represents the efficiency of resource sharing between the two endpoints. We use $N(u) := \{v \in V : (u, v) \in E\}$ to denote the set of neighbors for node $u \in V$. We use n and m to denote the number of nodes and edges in the graph G, respectively. For any integer i, we use $[i]$ to denote $\{1, 2, \ldots, i\}$.

The defender has a total resource of R that can be distributed to nodes in V. We use r_u to denote the *defending resource*[1] allocated to node u. Thus we have $\sum_{u\in V} r_u \le R$.

Definition 2.1 *(Pure strategy).* We use $\mathbf{r} = \{r_u\}_{u\in V}$ to denote a *pure strategy* and $\Omega_p(R) = \{\mathbf{r} \in [0,R]^V : \|\mathbf{r}\| = \sum_{u\in V} r_u \le R\}$[2] to denote the collection of pure strategies using resource R. When R is clear from the context, we use Ω_p to denote $\Omega_p(R)$.

We consider resource sharing in our model. That is, when node u is under attack, it can receive $w_{uv}\cdot r_v$ units of resource shared from each of its neighbors $v \in N(u)$.

Definition 2.2 *(Defending power).* Given pure strategy $\mathbf{r}$, the defending power of node u is defined as $\pi_u(\mathbf{r}) = r_u + \sum_{v\in N(u)} w_{uv}\cdot r_v$. We use $\boldsymbol{\pi}(\mathbf{r}) = (\pi_u(\mathbf{r}))_{u\in V}$ to denote defending powers of nodes.

Definition 2.3 *(Defending status).* Given a pure strategy $\mathbf{r}$, we use $\mathbf{x}(\mathbf{r}) \in \{0,1\}^V$ to denote the defending status of nodes under $\mathbf{r}$, where for each node u we have $x_u(\mathbf{r}) = 1$ if $\pi_u \ge \theta_u$, i.e., node u is well defended; and $x_u(\mathbf{r}) = 0$ if $\pi_u < \theta_u$, i.e., node u is not well defended.

Each pure strategy $\mathbf{r}$ has a unique defending status $\mathbf{x}(\mathbf{r})$ but different strategies can have the same defending status.

Definition 2.4 *(Defending result).* Given a pure strategy $\mathbf{r}$, when node $u \in V$ is under attack, the loss is given by $L_p(u,\mathbf{r}) = \alpha_u$ if $x_u(\mathbf{r}) = 0$; $L_p(u,\mathbf{r}) = 0$ otherwise. The defending result of strategy $\mathbf{r}$ is defined as the maximum loss due to an attack: $L_p(\mathbf{r}) = \max_{u\in V}\{L_p(u,\mathbf{r})\}$.

We use $\mathbf{r}^*$ to denote the optimal pure strategy, i.e., the pure strategy that has the minimum defending result $\mathbf{r}^* = \arg\min_{\mathbf{r}\in\Omega_p}\{L_p(\mathbf{r})\}$. The corresponding defending result is defined as $\mathsf{OPT}_p = L_p(\mathbf{r}^*)$.

Definition 2.5 *(Mixed strategy).* A mixed strategy is denoted by $(D,\mathbf{p})$, where $D \subseteq \Omega_p$ is a subset of pure strategies and $\mathbf{p}$ is a probability distribution over D. For each $\mathbf{r} \in D$, we use $p(\mathbf{r})$ to denote the probability that pure strategy $\mathbf{r}$ is used.

A mixed strategy is a randomized algorithm that applies pure strategies with certain probabilities. Note that $\sum_{\mathbf{r}\in D} p(\mathbf{r}) = 1$. We can also interpret $\mathbf{p}$ as a $|D|$ dimension vector with $\|\mathbf{p}\| = 1$. We use

$$\Omega_m(R) = \{(D,\mathbf{p}) : D \subseteq \Omega_p(R), \mathbf{p} \in [0,1]^{|D|}, \|\mathbf{p}\| = 1\}$$

to denote the collection of all mixed strategies using total resource R. When R is clear from the context, we use Ω_m to denote $\Omega_m(R)$.

Definition 2.6 *(Defending status of mixed strategy).* Given a mixed strategy $(D,\mathbf{p})$, we use $x_u(D,\mathbf{p}) = \sum_{\mathbf{r}\in D} p(\mathbf{r})\cdot x_u(\mathbf{r})$ to denote the defending status of node $u \in V$ under $(D,\mathbf{p})$. In other words, $x_u(D,\mathbf{p})$ is the probability that node u is well defended under mixed strategy $(D,\mathbf{p})$. We use $\mathbf{x}(D,\mathbf{p}) = (x_u(D,\mathbf{p}))_{u\in V} \in [0,1]^V$ to denote the defending status of $(D,\mathbf{p})$.

Definition 2.7 *(Defending result of mixed strategy).* Given mixed strategy $(D,\mathbf{p})$, we use $L_m(u,(D,\mathbf{p})) = (1 - x_u(D,\mathbf{p}))\cdot\alpha_u$ to denote the (expected) loss when node u is under attack. The defending result is defined as $L_m(D,\mathbf{p}) = \max_{u\in V}\{L_m(u,(D,\mathbf{p}))\}$.

We use $(D^*,\mathbf{p}^*)$ to denote the optimal mixed strategy, i.e., the mixed strategy with the minimum defending result. The corresponding defending result is defined as $\mathsf{OPT}_m = L_m(D^*,\mathbf{p}^*)$.

Next we define the fractional strategies. Technically, a fractional strategy is not a strategy, but a pure strategy equipped with a fractional valuation of defending loss. In the remaining of this paper, when a pure strategy is evaluated by its fractional loss, we call it a *fractional strategy*.

Definition 2.8 *(Fractional loss).* Given a pure strategy $\mathbf{r} \in \Omega_p$, we evaluate the fractional loss when node u is attacked by $L_f(u,\mathbf{r}) = (1 - \min\{\pi_u(\mathbf{r})/\theta_u, 1\})\cdot\alpha_u$. The fractional defending result is defined as $L_f(\mathbf{r}) = \max_{u\in V}\{L_f(u,\mathbf{r})\}$.

In a fractional strategy, if a node u has defending power π_u, then we assume that $\min\{\pi_u/\theta_u, 1\}$ fraction of the node is defended. Thus when node u is under attack, the loss is given by $(1 - \min\{\pi_u/\theta_u, 1\})\cdot\alpha_u$. We use $\tilde{\mathbf{r}}^*$ to denote the optimal fractional strategy, i.e., the strategy with minimum $L_f(\tilde{\mathbf{r}}^*)$. The corresponding defending result is defined as OPT_f. We use $\mathsf{OPT}_p(R), \mathsf{OPT}_m(R)$ and $\mathsf{OPT}_f(R)$ to denote the defending result of the optimal pure, mixed and fractional strategy using total resource R, respectively. When R is clear from the context, we simply use $\mathsf{OPT}_p, \mathsf{OPT}_m$ and OPT_f.

Next we use an example to illustrate the difference between mixed strategy and fractional strategy.

[1] As in [20,19], we assume the resource can be allocated arbitrarily in our model.

[2] Throughout this paper we use $\|\cdot\|$ to denote the L_1 norm of a vector.

Example 2.9. Consider an instance involving three targets, with their thresholds and values detailed in the table below. With a total resource of $R = 4$, the optimal mixed strategy applies each of the pure strategies $(3, 0, 1)$ and $(0, 3, 1)$ with a probability of $\frac{1}{2}$, resulting in a defending result of 1. In contrast, the compact representation $\left(\frac{15}{8}, \frac{15}{8}, \frac{1}{4}\right)$ yields a defending result of $\frac{3}{4}$, demonstrating that no mixed strategy can achieve this same defending result.

Target	a	b	c
Value	2	2	1
Threshold	3	3	1

The following lemma implies that the optimal fractional strategy has a defending result at most that of the optimal mixed strategy.

Lemma 2.10. *For any problem instance, we have* $OPT_p \geq OPT_m \geq OPT_f$.

Proof. Note that every pure strategy $\mathbf{r}$ is also a mixed strategy (with $D = \{\mathbf{r}\}$ and $p(\mathbf{r}) = 1$). Hence the first inequality trivially holds. In the following, we show that for any mixed strategy $(D, \mathbf{p})$, we can find a fractional strategy $\tilde{\mathbf{r}}$ using the same total resource R such that $L_m(D, \mathbf{p}) \geq L_f(\tilde{\mathbf{r}})$.

Let $\tilde{r}_u = \sum_{\mathbf{r} \in D} p(\mathbf{r}) \cdot r_u$ be the expected resource node u receives under $(D, \mathbf{p})$. Let $\tilde{\mathbf{r}} = (\tilde{r}_u)_{u \in V}$ be the resulting fractional strategy. Note that we have $\pi_u(\tilde{\mathbf{r}}) = \sum_{\mathbf{r} \in D} p(\mathbf{r}) \cdot \pi_u(\mathbf{r})$. Furthermore, $\|\tilde{\mathbf{r}}\| \leq R$, i.e., it uses a total resource at most R. Observe that when node u is under attack we have

$$\begin{aligned}
&L_m(u, (D, \mathbf{p})) = (1 - x_u(D, \mathbf{p})) \cdot \alpha_u \\
=&(1 - \textstyle\sum_{\mathbf{r} \in D} p(\mathbf{r}) \cdot x_u(\mathbf{r})) \cdot \alpha_u \\
\geq&(1 - \textstyle\sum_{\mathbf{r} \in D} p(\mathbf{r}) \cdot \min\{\pi_u(\mathbf{r})/\theta_u, 1\}) \cdot \alpha_u \\
\geq&(1 - \min\{\textstyle\sum_{\mathbf{r} \in D} p(\mathbf{r}) \cdot \pi_u(\mathbf{r})/\theta_u, 1\}) \cdot \alpha_u = L_f(u, \tilde{\mathbf{r}}).
\end{aligned}$$

It means that $L_m(D, \mathbf{p}) \geq L_f(\tilde{\mathbf{r}})$ since above relation holds for each node, which implies that $\mathsf{OPT}_m \geq \mathsf{OPT}_f$. □

3. Computation of strategies

In this section we consider the computation of the optimal pure, mixed and fractional strategies, and also analyze some properties regarding the optimal defending results of different strategies.

3.1. Optimal pure and fractional strategy

We remark that our model is equal to the "single threshold" model of [20]. We thus use their algorithm (that runs in polynomial time) to compute an optimal pure strategy. Roughly speaking, in their algorithm a target defending result α is fixed and the goal is to decide whether it is possible to defend all nodes $A(\alpha) = \{u \in V : \alpha_u > \alpha\}$ with value larger than α. For every fixed α the above decision problem can be solved by solving a feasibility LP with constraints $\sum_{u \in V} r_u \leq R$ and $r_u + \sum_{v \in N(u)} w_{uv} \cdot r_v \geq \theta_u$ for every $u \in A(\alpha)$. Combining the above subroutine with a binary search on $\alpha \in \{\alpha_u\}_{u \in V} \cup \{0\}$ yields a polynomial time algorithm for computing the optimal pure strategy, i.e., with the minimum achievable defending result α.

The computation of the optimal fractional strategy can be done efficiently by solving the following linear program $(\mathsf{LP}_f(R))$, where we introduce a variable r_u for each node $u \in V$ that represents the resource u receives, and a variable L for the defending result.

$$\begin{aligned}
(\mathsf{LP}_f(R)) \qquad &\text{minimize} && L \\
&\text{subject to} && \textstyle\sum_{u \in V} r_u \leq R, \\
&(1 - (r_u + \textstyle\sum_{v \in N(u)} w_{uv} \cdot r_v)/\theta_u) \cdot \alpha_u \leq L, && \forall u \in V
\end{aligned}$$

By solving the above LP we get the optimal fractional strategy $\tilde{\mathbf{r}}^*$, whose defending result OPT_f is the optimal objective of the LP. From Lemma 2.10, we have $\mathsf{OPT}_f \leq \mathsf{OPT}_m$, i.e., we can use OPT_f as a lower bound for the defending result of the optimal mixed strategy (which is NP-hard to compute, as we will show later). In the following we show that the optimal objective of the above LP is a convex function of the total resource R.

Lemma 3.1 *(Convexity). Given resource* R_1 *and* R_2*, we have*

$$OPT_f(R_1) + OPT_f(R_2) \geq 2 \cdot OPT_f\left(\tfrac{1}{2}(R_1 + R_2)\right).$$

Proof. Let $\tilde{\mathbf{r}}_1^*$ and $\tilde{\mathbf{r}}_2^*$ be the optimal fractional strategies given resource R_1 and R_2, respectively. Note that $(\tilde{\mathbf{r}}_1^*, \mathsf{OPT}_f(R_1))$ and $(\tilde{\mathbf{r}}_2^*, \mathsf{OPT}_f(R_2))$ are feasible solutions to $(\mathsf{LP}_f(R_1))$ and $(\mathsf{LP}_f(R_2))$, respectively. Let $\tilde{\mathbf{r}} = \frac{1}{2} \cdot (\tilde{\mathbf{r}}_1^* + \tilde{\mathbf{r}}_2^*)$. In the following we show that $(\tilde{\mathbf{r}}, \frac{1}{2} \cdot (\mathsf{OPT}_f(R_1) + \mathsf{OPT}_f(R_2)))$ is a feasible solution to $(\mathsf{LP}_f(\frac{1}{2}(R_1 + R_2)))$. The first constraint of the LP trivially holds because $\|\tilde{\mathbf{r}}\| = \frac{1}{2} \cdot (\|\tilde{\mathbf{r}}_1^*\| + \|\tilde{\mathbf{r}}_2^*\|) \le \frac{1}{2}(R_1 + R_2)$. By the feasibility of $(\tilde{\mathbf{r}}_1^*, \mathsf{OPT}_f(R_1))$ and $(\tilde{\mathbf{r}}_2^*, \mathsf{OPT}_f(R_2))$, we have the following relations:

$$(1 - \pi_u(\tilde{\mathbf{r}}_1^*)/\theta_u) \cdot \alpha_u \le \mathsf{OPT}_f(R_1), \qquad \forall u \in V$$
$$(1 - \pi_u(\tilde{\mathbf{r}}_2^*)/\theta_u) \cdot \alpha_u \le \mathsf{OPT}_f(R_2), \qquad \forall u \in V.$$

Combining the two sets of inequalities we get

$$(1 - \tfrac{1}{2}(\pi_u(\tilde{\mathbf{r}}_1^*) + \pi_u(\tilde{\mathbf{r}}_2^*))/\theta_u) \cdot \alpha_u \le \tfrac{1}{2} \cdot (\mathsf{OPT}_f(R_1) + \mathsf{OPT}_f(R_2)), \forall u \in V.$$

Since we have $\pi_u(\tilde{\mathbf{r}}) = \frac{1}{2}(\pi_u(\tilde{\mathbf{r}}_1^*) + \pi_u(\tilde{\mathbf{r}}_2^*))$, we conclude that $(\tilde{\mathbf{r}}, \frac{1}{2}(\mathsf{OPT}_f(R_1) + \mathsf{OPT}_f(R_2)))$ is a feasible solution to $(\mathsf{LP}_f(\frac{1}{2}(R_1 + R_2)))$. Consequently the optimal objective of the LP has

$$\mathsf{OPT}_f(\tfrac{1}{2}(R_1 + R_2)) \le \tfrac{1}{2} \cdot (\mathsf{OPT}_f(R_1) + \mathsf{OPT}_f(R_2)).$$

Rearranging the inequality concludes the proof. □

3.2. Hardness for computing mixed strategies

We have shown that the optimal pure and fractional strategies can be computed efficiently. Unfortunately, we show that computing the optimal mixed strategy is NP-hard, even in the *isolated model*, i.e., when $w_{uv} = 0$ for all $(u, v) \in E$.

Theorem 3.2. *Unless $P = NP$, there does not exist any polynomial time algorithm that given a graph $G(V, E)$ and resource R computes the optimal mixed strategy, even under the isolated model.*

Proof. We prove the hardness result by a reduction from the Even Partition problem, which is known to be NP-complete [37]. Given a set of numbers $A = \{a_1, a_2, ..., a_n\}$, the problem is to decide whether A can be partitioned into two subsets of equal sum. Given set A, we construct the instance of the defending problem as follows. Let $G(V, E)$ be a graph with $V = [n]$ and $E = \emptyset$. For each node $i \in V$ we set $\theta_i = a_i$ and $\alpha_i = 1$. We set the total resource $R = \frac{1}{2}\sum_{i \in A} a_i$.

Obviously if A can be partitioned into two sets A_1 and A_2 of equal sum, then both of them have sum equals to R. Then we can define two pure strategies: the first strategy allocates resource $r_i = \theta_i$ for each $i \in A_1$; the second one allocates resource $r_i = \theta_i$ for each $i \in A_2$. Then we define a mixed strategy that applies each of these two strategies with probability 0.5. It is easy to check that the defending result is 0.5 since the defending status of each node is 0.5. Hence if A has an even partition, we have $\mathsf{OPT}_m(R) \le 0.5$.

On the other hand, we show that if A does not have an even partition, then $\mathsf{OPT}_m(R) > 0.5$. Let R' be the maximum sum of numbers in A that is at most R. Since A does not have an even partition, we have $R' < R$. Moreover, in every pure strategy the total threshold of nodes that are well defended is at most R'. In other words, for every $\mathbf{r} \in \Omega_p(R)$, there is a corresponding $\mathbf{r}' \in \Omega_p(R')$ with $\mathbf{x}(\mathbf{r}') = \mathbf{x}(\mathbf{r})$. Thus we have $\mathsf{OPT}_m(R) = \mathsf{OPT}_m(R')$. Observe that since $R' < R = \frac{1}{2} \cdot \sum_{i \in V} \theta_i$, in any fractional strategy using total resource R', there must exist a node $i \in V$ with $r_i < 0.5 \cdot \theta_i$. Consequently, we have $\mathsf{OPT}_f(R') > 0.5$. Finally, by Lemma 2.10, we have $\mathsf{OPT}_m(R) = \mathsf{OPT}_m(R') \ge \mathsf{OPT}_f(R') > 0.5$, as claimed.

In conclusion, we have $\mathsf{OPT}_m(R) \le 0.5$ if and only if A admits an even partition. Since the reduction is in polynomial time, we know that the computation of the optimal mixed strategy is NP-hard. □

3.3. A strong upper bound for isolated model

While computing the optimal mixed strategy is NP-hard, we can use $\mathsf{OPT}_f(R)$ to give a lower bound on $\mathsf{OPT}_m(R)$. In other words, if a mixed strategy has a defending result close to $\mathsf{OPT}_f(R)$, then it is close-to-optimal. However, if the lower bound is loose, then no such mixed strategy exists. Therefore, it is crucial to know whether this lower bound is tight. In this section, we show that in the isolated model, we can give a strong upper bound on $\mathsf{OPT}_m(R)$, which shows that $\mathsf{OPT}_f(R)$ is an almost tight lower bound when R is large.

Theorem 3.3. *In the isolated model, given any instance $G(V, E)$ and a total resource R, we have*

$$OPT_m(R) \le OPT_f(R - \theta_{max}),$$

where $\theta_{max} = \max_{u \in V}\{\theta_u\}$ is the maximum threshold of the nodes.

Before presenting the proof, we remark that by convexity of the function $\mathsf{OPT}_f(\cdot)$, we have

$$\mathsf{OPT}_f(R - \theta_{\max}) \le \frac{R - \theta_{\max}}{R} \cdot \mathsf{OPT}_f(R) + \frac{\theta_{\max}}{R} \cdot \mathsf{OPT}_f(0)$$

$$= \mathsf{OPT}_f(R) + \frac{\theta_{\max}}{R} \cdot \left(\max_{u \in V} \{\alpha_u\} - \mathsf{OPT}_f(R) \right).$$

In other words, when $R \gg \theta_{\max}$, $\mathsf{OPT}_f(R)$ and $\mathsf{OPT}_f(R - \theta_{\max})$ have very similar values. Hence combining Lemma 2.10 and Theorem 3.3, we have strong upper and lower bounds on $\mathsf{OPT}_m(R)$ when $R \gg \theta_{\max}$. Furthermore, we remark that following our analysis, it can be verified that if $\theta_u = \theta_{\max}$ for all nodes $u \in V$ and R is divisible by $\theta_{\max}$, then we can prove the stronger result $\mathsf{OPT}_m(R) = \mathsf{OPT}_f(R)$. Moreover, there exists a mixed strategy $(D, \mathbf{p})$ with $|D| = O(n^2)$ that achieves this defending result. In other words, our analysis also reproduces the result of [12]. We prove Theorem 3.3 by showing the following lemma.

Lemma 3.4. *Given any vector $\mathbf{f} \in [0,1]^V$ with $\sum_{u \in V} f_u \cdot \theta_u \le R - \theta_{max}$, we can compute in polynomial time a mixed strategy $(D, \mathbf{p}) \in \Omega_m(R)$ with $|D| = O(n^2)$ such that $\mathbf{x}(D, \mathbf{p}) = \mathbf{f}$.*

In particular, let $\tilde{\mathbf{r}}^*$ be the optimal fractional strategy using resource $R - \theta_{\max}$. Note that in the isolated model we have $\pi_u(\tilde{\mathbf{r}}^*) = \tilde{r}^*_u$. Let $\mathbf{f} \in [0,1]^V$ be defined by $f_u = \min\{\tilde{r}^*_u / \theta_u, 1\}$, for all $u \in V$. That is, f_u is the fraction node u is defended in the fractional strategy. Then $\mathbf{f}$ satisfies the condition of Lemma 3.4, and hence there exists a mixed strategy $(D, \mathbf{p}) \in \Omega_m(R)$ with $\mathbf{x}(D, \mathbf{p}) = \mathbf{f}$. Hence we have

$$\begin{aligned} &\mathsf{OPT}_m(R) \le L_m(D, \mathbf{p}) = \max_{u \in V} \{(1 - x_u(D, \mathbf{p})) \cdot \alpha_u\} \\ &= \max_{u \in V} \{(1 - f_u) \cdot \alpha_u\} = L_f(\tilde{\mathbf{r}}^*) = \mathsf{OPT}_f(R - \theta_{\max}). \end{aligned}$$

3.4. Proof of Lemma 3.4

We prove the lemma by giving a polynomial time algorithm that given the vector $\mathbf{f}$ computes the mixed strategy $(D, \mathbf{p})$ with the claimed properties. For uniform thresholds (when $\theta_u = \theta_{\max}$ for all nodes u and R is divisible by $\theta_{\max}$), we can have the same guarantee when $\sum_{u \in V} f_u \cdot \theta_u \le R$. Intuitively, we prove the lemma by showing that for any subset of nodes, we can use $O(n)$ pure strategies to increase their defending status by the same amount. It can be shown that any vector $\mathbf{f} \in [0,1]^V$ can be decomposed into $O(n)$ "canonical" vectors, where a vector is canonical if all its non-zero dimensions have the same value. Therefore by expressing each canonical component of $\mathbf{f}$ using $O(n)$ pure strategies, the total number of strategies used is $O(n^2)$.

For convenience of discussion, we first introduce the following notations.

Notations. In the isolated model, it makes no sense to allocate resource $r_u \in (0, \theta_u)$ to a node u. Thus we only consider pure strategies $\mathbf{r}$ with $r_u \in \{0, \theta_u\}$ for all $u \in V$, and let $\widetilde{\Omega}_p(R)$ be the collection of such pure strategies using total resource at most R. For a vector $\mathbf{f} \in [0,1]^V$, we use $V_{\max}(\mathbf{f}) = \{u \in V : f_u = \max_{v \in V} \{f_v\}\}$ to denote the set of nodes u with maximum f_u, and $V_0(\mathbf{f}) = \{u \in V : f_u = 0\}$. In addition, given vector $\mathbf{f}$, we define $\mathrm{MaxTop}(\mathbf{f}) \subseteq V$ to be a set of nodes with total threshold at most R as follows. We initialize $\mathrm{MaxTop}(\mathbf{f}) \leftarrow \emptyset$ and then greedily include nodes $u \in V \setminus \mathrm{MaxTop}(\mathbf{f})$ with maximum f_u value (break ties by the index of nodes) into $\mathrm{MaxTop}(\mathbf{f})$ as long as $f_u > 0$ and the resulting set of nodes has total threshold at most R.

Algorithm 1: MaxTop($\mathbf{f}$).

Input: the fractional strategy $\mathbf{f}$
initialize $M \leftarrow \emptyset$;
while $M \neq V \setminus V_0(\mathbf{f})$ **do**
  $u \leftarrow \arg\max_{v \in V \setminus M} \{f_v\}$;
  if $\theta_u + \sum_{v \in M} \theta_v \le R$ **then**
    $M \leftarrow M \cup \{u\}$;
  else
    return M
Output: a set of nodes with total threshold at most R.

Notice that there exists a pure strategy $\mathbf{r} \in \widetilde{\Omega}_p(R)$ that defends (and only defends) the nodes in $\mathrm{MaxTop}(\mathbf{f})$ simultaneously. Furthermore, unless $\mathrm{MaxTop}(\mathbf{f})$ contains all nodes with non-zero f values, i.e., $M = V \setminus V_0(\mathbf{f})$, the total resource $\mathbf{r}$ uses is $\|\mathbf{r}\| = \sum_{u \in \mathrm{MaxTop}(\mathbf{f})} \theta_u > R - \theta_{\max}$. Note that for uniform thresholds, we have $\|\mathbf{r}\| = |\mathrm{MaxTop}(\mathbf{f})| \cdot \theta_{\max} = R$, which means that all resources are used up in each pure strategy $\mathbf{r} \in \widetilde{\Omega}_p(R)$.

3.4.1. Overview

Let $\mathbf{f}^{(0)}$ be the vector $\mathbf{f}$ given in Lemma 3.4. Recall that our goal is to find $O(n^2)$ pure strategies $D \subseteq \widetilde{\Omega}_p(R)$ and associate a probability $p(\mathbf{r})$ to each $\mathbf{r} \in D$ satisfying[3]

[3] Formally, we need equality here. However, given any mixed strategy whose total probability is $1 - \epsilon$, we can add a dummy pure strategy (that allocates 0 resource to every node) with probability ϵ without changing the defending result.

$$\sum_{\mathbf{r}\in D} p(\mathbf{r}) \le 1, \tag{1}$$

such that $\mathbf{x}(D,\mathbf{p}) = \mathbf{f}^{(0)}$. We implement this goal by progressively including new pure strategies (with certain probabilities) into D as long as

$$\mathbf{x}(D,\mathbf{p}) \le \mathbf{f}^{(0)}. \tag{2}$$

In particular, we let $\mathbf{f} = \mathbf{f}^{(0)} - \mathbf{x}(D,\mathbf{p})$ be the *residual vector*, which will be dynamically updated when we include new pure strategies into D. The goal is to eventually decrease $\mathbf{f}$ to the all-zero vector $\mathbf{0}$. In such case our algorithm terminates and outputs the mixed strategy $(D,\mathbf{p})$. Our algorithm works in iterations. In each iteration, the algorithm includes $O(n)$ pure strategies into D,[4] and guarantees that the inclusion of new strategies increases $|V_{\max}(\mathbf{f})| + |V_0(\mathbf{f})|$ by at least one. It can also be verified that throughout the whole algorithm $|V_{\max}(\mathbf{f})|$ and $|V_0(\mathbf{f})|$ never decrease. The algorithm terminates when $|V_{\max}(\mathbf{f})| + |V_0(\mathbf{f})| > n$, in which case we have $V_{\max}(\mathbf{f}) \cap V_0(\mathbf{f}) \neq \emptyset$, which implies $V_{\max}(\mathbf{f}) = V_0(\mathbf{f}) = V$, i.e., $\mathbf{f}$ is an all-zero vector.

Phase A. A natural idea is to include the pure strategy $\mathbf{r}$ that defends the nodes MaxTop($\mathbf{f}$), i.e., those with largest f values, into D and give it an appropriate probability satisfying conditions (1) and (2). In particular, suppose $V_{\max}(\mathbf{f}) \subseteq \text{MaxTop}(\mathbf{f})$. We continuously increase $p(\mathbf{r})$ (which decreases f_u for all $u \in \text{MaxTop}(\mathbf{f})$ at the same rate) until one of the following two events happens

(a) the maximum f value of nodes in MaxTop($\mathbf{f}$) is the same as $\max_{v\notin \text{MaxTop}(\mathbf{f})}\{f_v\}$; or
(b) $f_u = 0$ for some $u \in \text{MaxTop}(\mathbf{f})$.

In Case-(a), we increase $|V_{\max}(\mathbf{f})|$ by at least one; in Case-(b), we increase $|V_0(\mathbf{f})|$ by at least one without decreasing $|V_{\max}(\mathbf{f})|$. In either case, we can finish the iteration with $|V_{\max}(\mathbf{f})| + |V_0(\mathbf{f})|$ increased by at least one. The subtle case is when $V_{\max}(\mathbf{f}) \subsetneq \text{MaxTop}(\mathbf{f})$. In such case, the strategy $\mathbf{r}$ (that defends nodes in MaxTop($\mathbf{f}$)) falls short of defending all nodes in $V_{\max}(\mathbf{f})$. As a consequence, we can only have $p(\mathbf{r}) = 0$ because any $p(\mathbf{r}) > 0$ may result in a decrease in $|V_{\max}(\mathbf{f})|$. Hence, when this happens, our algorithm enters Phase B.

Phase B. Observe that $V_{\max}(\mathbf{f}) \subsetneq \text{MaxTop}(\mathbf{f})$ is equivalent to $\sum_{u\in V_{\max}(\mathbf{f})} \theta_u > R$. We show that in this case, we can find $O(n)$ pure strategies and associate a probability to each of them such that after including these strategies to D, we can decrease f_u for each $u \in V_{\max}(\mathbf{f})$ by

$$\epsilon := \max_{u\in V_{\max}(\mathbf{f})}\{f_u\} - \max_{v\notin V_{\max}(\mathbf{f})}\{f_v\}. \tag{3}$$

The following lemma is the key to find these $O(n)$ pure strategies.

Lemma 3.5. *Given any vector $\mathbf{t} \in \{0,\epsilon\}^V$ with $\sum_{u\in V_{max}(\mathbf{t})} \theta_u > R$, we can find $O(|V_{max}(\mathbf{t})|)$ pure strategies $T \subseteq \widetilde{\Omega}_p(R)$ such that $\|\mathbf{r}\| > R - \theta_{max}, \forall \mathbf{r} \in T$. For uniform thresholds, we further have $\|\mathbf{r}\| = R$. Moreover, there exists an integer $c > 0$ such that*

$$\mathbf{t} = \tfrac{\epsilon}{c} \cdot \sum_{\mathbf{r}\in T} \mathbf{x}(\mathbf{r}).$$

We define $\mathbf{t}$ as $t_u = 0$ for all $u \notin V_{\max}(\mathbf{f})$; $t_u = \epsilon$ for all $u \in V_{\max}(\mathbf{f})$, where ϵ is as defined in (3). Then by Lemma 3.5, we can find $O(|V_{\max}(\mathbf{t})|) = O(n)$ pure strategies T with the above properties. By assigning probability $p(\mathbf{r}) = \epsilon/c$ for each $\mathbf{r} \in T$ and including these strategies into D, we can decrease f_u for $u \in V_{\max}(\mathbf{f})$ by ϵ. As a consequence, including these new pure strategies increases $|V_{\max}(\mathbf{f})|$ by at least one. Hence when this iteration finishes we have $|V_{\max}(\mathbf{f})| + |V_0(\mathbf{f})|$ increased by at least one. Observe that in the next iteration we also have $V_{\max}(\mathbf{f}) \subsetneq \text{MaxTop}(\mathbf{f})$. In other words, the algorithm stays in Phase B until it terminates.

3.4.2. Proof of Lemma 3.5

Let $\mathbf{t} \in \{0,\epsilon\}^V$ be the vector given in Lemma 3.5, and $k = |V_{\max}(\mathbf{t})|$ be the number of non-zero coordinates. To prove Lemma 3.5, we will show that there exists $T \subseteq \widetilde{\Omega}_p(R)$ satisfying following conditions:

(a) each $\mathbf{r} \in T$ defends only nodes in $V_{\max}(\mathbf{t})$, i.e., $r_u = 0$ if $u \notin V_{\max}(\mathbf{t})$;
(b) each $\mathbf{r} \in T$ uses total resource $\|\mathbf{r}\| > R - \theta_{\max}$;
(c) there exists an integer c such that for every node $u \in V_{\max}(\mathbf{t})$, the number of pure strategies in which u is well defended is $|\{\mathbf{r} \in T : r_u = \theta_u\}| = c$;
(d) $|T| \le k$.

Since strategies in T defend only nodes in $V_{\max}(\mathbf{t})$ and each node in $V_{\max}(\mathbf{t})$ is defended by the same number of pure strategies, we have $\mathbf{t} = \frac{\epsilon}{c} \cdot \sum_{\mathbf{r}\in T} \mathbf{x}(\mathbf{r})$, as claimed in Lemma 3.5. We remark that condition (a) and (b) are relatively easy to satisfy. The tricky part is to satisfy condition (c) using only k strategies (condition (d)). We accomplish the mission by proposing the following algorithm.

[4] As we may include the same pure strategy into D multiple times in different iterations, D would be a multi-set of pure strategies, in which if a pure strategy appears several times, they are regarded as different strategies and can have different probabilities.

In the following, we present the ideas for computing T.

Let $\{u_1, \ldots, u_k\}$ be the nodes in $V_{\max}(\mathbf{t})$, indexed by their IDs. Suppose $\text{MaxTop}(\mathbf{t}) = \{u_1, \ldots, u_i\}$, where $i < k$. That is, $\sum_{j=1}^{i} \theta_{u_j} \le R$ but $\sum_{j=1}^{i+1} \theta_{u_j} > R$. Then we first include the strategy that defends node in $\text{MaxTop}(\mathbf{t})$ and try to find other strategies to defend the remaining nodes $\{u_{i+1}, \ldots, u_k\}$. Now suppose that $\sum_{j=i+1}^{k} \theta_{u_j} \le R - \theta_{\max}$. Then the pure strategy $\mathbf{r}$ that defends only nodes in $\{u_{i+1}, \ldots, u_k\}$ does not satisfy condition (b). To ensure condition (b) holds, we include nodes $\{u_1, u_2, \ldots\}$ into the set of nodes to be defended by $\mathbf{r}$, until we have $\|\mathbf{r}\| > R - \theta_{\max}$ (for uniform thresholds, we have $\|\mathbf{r}\| = R$). In particular, Algorithm 2 computes the maximal set of nodes (starting from u_i) to be defended, and also returns the end position j, i.e., u_{j-1} is defended but u_j is not. Suppose (M, j) is returned by $\text{CycleMaxTop}(\mathbf{t}, i)$. The main idea is to include the strategy that defends nodes in M, and then recursively call $\text{CycleMaxTop}(\mathbf{t}, j)$ to compute the next strategy. If for some call of $\text{CycleMaxTop}(\mathbf{t}, i)$, the returned end position $j = 1$, then we know that the pure strategies we have computed thus far defend all nodes the same number of times. Unfortunately, we cannot guarantee that this will happen, let alone guaranteeing this to happen in $O(k)$ rounds.

Fortunately, we have the following important observation. Every time when we call the function $\text{CycleMaxTop}(\mathbf{t}, i)$, we check whether such a call (with the same parameters $\mathbf{t}$ and i) has been made before. If yes, then the set of pure strategies computed since the first call to $\text{CycleMaxTop}(\mathbf{t}, i)$ (inclusive) till the second call (exclusive) must have defended all nodes in $V_{\max}(\mathbf{t})$ the same number of times: the first strategy defends a sequence of nodes starting from node u_i, and the last strategy defends a sequence of nodes ending at node u_{i-1}. In such case we extract this subset of pure strategies and return it as the desired set T. We summarize the steps in Algorithm 3.

Algorithm 2: $\text{CycleMaxTop}(\mathbf{t}, i)$.

```
1  initialize M ← ∅ ; // suppose V_max(t) = {u_1, ..., u_k} ;
2  while ∑_{u∈M} θ_u ≤ R − θ_max do
3      M ← M ∪ {u_i} ;
4      i ← 1 + (i mod k) ;
5  return (M, i)
```

Algorithm 3: $\text{FindT}(\mathbf{t})$.

```
   Input: V_max(t)
1  suppose V_max(t) = {u_1, ..., u_k} ;
2  initialize T ← ∅ and i ← 1 ;
3  while True do
4      (M, i) ← CycleMaxTop(t, i) ;
5      let r be defined as follows:
6      r_u = θ_u if u ∈ M and r_u = 0 otherwise ;
7      if r ∈ T then
8          remove all strategies in T that are included before r ;
9          pick an arbitrary u ∈ V_max(t), and set c ← |{r ∈ T : r_u = θ_u}| ;
10         return (T, c) ;
11     else
12         T ← T ∪ {r}
   Output: a set of strategy T that defends all nodes in V_max(t) with same number of times c.
```

Proof of Lemma 3.5. As argued above, it suffices to show that the computed set of pure strategies meet conditions (a) - (d). By the way the strategies are generated conditions (a) and (b) are easily satisfied. Condition (c) is satisfied because when we observe that function $\text{CycleMaxTop}(\mathbf{t}, i)$ is called for the second time with the same parameters, we keep only the strategies they are computed between these two calls. As shown above, these pure strategies defend all nodes in $V_{\max}(\mathbf{t})$ the same number of times. Finally, condition (d) is satisfied because we call the function $\text{CycleMaxTop}(\mathbf{t}, i)$ only for $i \in [k]$. Thus within $k + 1$ calls we must have found two calls with the same input parameter i, in which case Algorithm 3 terminates and outputs at most k strategies (line 7 - 9). □

3.4.3. The complete algorithm

We summarize the steps of our algorithm in Algorithm 4, which takes as input the nodes V (where each $u \in V$ has threshold θ_u), a resource bound R and a vector $\mathbf{f}^{(0)}$, and outputs a mixed strategy with properties stated in Lemma 3.4.

Each while loop of Algorithm 4 correspond to one iteration of our algorithm. In Particular, line 4 - 9 correspond to Phase A of the algorithm, during which we add one new pure strategy in each iteration; line 10 - 15 correspond to Phase B of the algorithm, which is called only if $V_{\max}(\mathbf{f}) \subsetneq \text{MaxTop}(\mathbf{f})$. In such case, we let $\mathbf{t}$ be defined as we stated in Section 3.4.1, and call the sub-routine $\text{FindT}(\mathbf{t})$ (the detailed description of the algorithm is included in the appendix) to compute the set of pure strategies $T \subseteq \widetilde{\Omega}_p(R)$ and the constant c as stated in Lemma 3.5. We include the pure strategies in T into D, and give each of them probability ϵ/c, which finishes the iteration.

Algorithm 4: Compute Mixed Strategy.

Input: V, $\{\theta_u\}_{u\in V}$, R, and $\mathbf{f}^{(0)} \in [0,1]^V$
1 $D \leftarrow \emptyset$, $\mathbf{f} \leftarrow \mathbf{f}^{(0)}$;
2 **while** $\|\mathbf{f}\| \neq \mathbf{0}$ **do**
3 $M \leftarrow \text{MaxTop}(\mathbf{f})$;
4 **if** $V_{max}(\mathbf{f}) \subseteq M$ **then**
5 let $\mathbf{r}$ be defined as follows:
6 $r_u = \theta_u$ if $u \in M$ and $r_u = 0$ otherwise ;
7 $D \leftarrow D \cup \{\mathbf{r}\}$;
8 $p(\mathbf{r}) \leftarrow \min\{\max_{u\in M}\{f_u\} - \max_{v\notin M}\{f_v\}, \min_{u\in M}\{f_u\}\}$;
9 update $\mathbf{f} \leftarrow \mathbf{f} - p(\mathbf{r}) \cdot \mathbf{x}(\mathbf{r})$;
10 **else**
11 $\epsilon \leftarrow \max_{u\in V_{\max}(\mathbf{f})}\{f_u\} - \max_{v\notin V_{\max}(\mathbf{f})}\{f_v\}$;
12 let $\mathbf{t}$ be defined as $t_u = \epsilon$ if $u \in V_{\max}(\mathbf{f})$ and $t_u = 0$ otherwise ;
13 $(T, c) \leftarrow \text{FindT}(\mathbf{t})$;
14 $D \leftarrow D \cup T$;
15 set $p(\mathbf{r}) \leftarrow \epsilon/c$ for all $\mathbf{r} \in T$ update $\mathbf{f} \leftarrow \mathbf{f} - \epsilon/c \cdot \sum_{\mathbf{r}\in T} \mathbf{x}(\mathbf{r})$;
16 **return** $(D, \mathbf{p})$
Output: The mixed strategy $(D, \mathbf{p})$

3.4.4. Analysis

We first prove the correctness of our algorithm, i.e., the mixed strategy $(D, \mathbf{p})$ returned by Algorithm 4 satisfies

(1) $\mathbf{x}(D, \mathbf{p}) = \mathbf{f}^{(0)}$;
(2) $D = O(n^2)$;
(3) $\sum_{\mathbf{r}\in D} p(\mathbf{r}) \leq 1$.

The first condition is easy to show because our algorithm always guarantees that $\mathbf{x}(D, \mathbf{p}) \leq \mathbf{f}^{(0)}$, and terminates only if $\mathbf{f} = \mathbf{f}^{(0)} - \mathbf{x}(D, \mathbf{p})$ is an all-zero vector. Following the arguments we have presented, in each iteration of Phase A we include one pure strategy into D; in each iteration of Phase B we include $O(n)$ pure strategies into D (by Lemma 3.5). Since each iteration increases $|V_{\max}(\mathbf{f})| + |V_0(\mathbf{f})|$ by at least one and our algorithm terminates when $|V_{\max}(\mathbf{f})| + |V_0(\mathbf{f})| > n$ (in which case we have $\mathbf{f} = \mathbf{0}$), we conclude that there are at most n iterations. Hence $|D| = O(n^2)$.

Next we show that $\sum_{\mathbf{r}\in D} p(\mathbf{r}) \leq 1$. We analyze total probability in two cases, depending on whether algorithm ever enters Phase B.

Let $u \in V_{\max}(\mathbf{f}^{(0)})$ be an arbitrary node with maximum f_u in $\mathbf{f}^{(0)}$. Throughout the whole algorithm, we can guarantee $u \in V_{\max}(\mathbf{f})$ for any $\mathbf{f}$, because we never decrease f_u to a value that is lower than the second largest f value. Hence if Algorithm 4 never enters Phase B, then we have

$$\sum_{\mathbf{r}\in D} p(\mathbf{r}) = f_u \leq 1.$$

Now suppose that Algorithm 4 terminates at Phase B. The important observation here is that in such case we have $\sum_{u\in M} \theta_u > R - \theta_{\max}$ for every M that is returned by $\text{MaxTop}(\mathbf{f})$ in line 3, because $\sum_{u\in M} \theta_u \leq R - \theta_{\max}$ happens only if $\text{MaxTop}(\mathbf{f}) = V \setminus V_0(\mathbf{f})$, in which case the algorithm never enters Phase B. Consequently for each $\mathbf{r} \in D$ we have $\|\mathbf{r}\| > R - \theta_{\max}$ (for uniform thresholds we have $\|\mathbf{r}\| = R$). Recall that $\mathbf{f}^{(0)}$ is defined such that for some $\tilde{\mathbf{r}}^* \in \Omega_p(R - \theta_{\max})$, $f_u^{(0)} = \min\{\tilde{r}_u^*/\theta_u, 1\}$ for all $u \in V$. Also recall that for each $u \in V$ we have

$$\sum_{\mathbf{r}\in D} (p(\mathbf{r}) \cdot r_u) = \theta_u \cdot x_u(D, \mathbf{p}) = \theta_u \cdot f_u^{(0)}.$$

It follows that

$$\sum_{u\in V} \sum_{\mathbf{r}\in D} (p(\mathbf{r}) \cdot r_u) = \sum_{u\in V} (\theta_u \cdot f_u^{(0)}) \leq \sum_{u\in V} \tilde{r}_u^* \leq R - \theta_{\max}. \tag{4}$$

On the other hand, we have

$$\sum_{u\in V} \sum_{\mathbf{r}\in D} (p(\mathbf{r}) \cdot r_u) = \sum_{\mathbf{r}\in D} (p(\mathbf{r}) \cdot \|\mathbf{r}\|) > (R - \theta_{\max}) \cdot \sum_{\mathbf{r}\in D} p(\mathbf{r}). \tag{5}$$

Combining (4) and (5), we have $\sum_{\mathbf{r}\in D} p(\mathbf{r}) < 1$.

For uniform thresholds we have

$$\sum_{u\in V} \sum_{\mathbf{r}\in D} (p(\mathbf{r}) \cdot r_u) = \sum_{u\in V} (\theta_u \cdot f_u^{(0)}) = \sum_{u\in V} \tilde{r}_u^* = R, \tag{6}$$

and

$$\sum_{u\in V}\sum_{\mathbf{r}\in D}(p(\mathbf{r})\cdot r_u)=\sum_{\mathbf{r}\in D}(p(\mathbf{r})\cdot\|\mathbf{r}\|)=R\cdot\sum_{\mathbf{r}\in D}p(\mathbf{r}). \tag{7}$$

Combining (6) and (7), we have $\sum_{\mathbf{r}\in D}p(\mathbf{r})=1$.

Complexity. Now we analyze the complexity of Algorithm 4. It is easy to check that $V_{\max}(\mathbf{f})$, $V_0(\mathbf{f})$, MaxTop($\mathbf{f}$) and CycleMaxTop($\mathbf{t},i$) can be computed in $O(n\log n)$ time. From the proof of Lemma 3.5, we know that Algorithm 3 executes in $O(n)$ rounds. Thus each call to FindT($\mathbf{t}$) finishes in $O(n^2\log n)$ time. Finally, since there are $O(n)$ iterations, and in each iteration FindT($\mathbf{t}$) is called at most once, the total complexity of Algorithm 4 is $O(n^3\log n)$.

3.5. Mixed strategy with resource sharing

As we have shown above, in the isolated model we can give a strong upper bound $\mathsf{OPT}_m(R)$ by $\mathsf{OPT}_f(R-\theta_{\max})$. It would be natural to ask whether similar upper bounds hold under the non-isolated model, i.e., when defending resource can be shared between neighboring nodes. Unfortunately, we show that when resource sharing is allowed, we do not have such guarantees, even if we allow the mixed strategy to use several times more resource than the fractional strategy.

Lemma 3.6. *For any constant $\beta>1$, there exists an instance for which $OPT_m(\beta\cdot R)>OPT_f(R)$.*

Proof. Consider a complete bipartite graph $G(U\cup V,E)$, where $|U|=2\beta\cdot R, |V|=4\beta^2\cdot R$ and all edges $(u,v)\in E=U\times V$ have the same weight $w_{uv}=1/|U|$. Let $\theta_u=\alpha_u=1$ for all $u\in U\cup V$.

Observe that there exists a fractional strategy (using total resource R) that allocates $1/(2\beta)$ resource to each of the nodes in U, under which every node in U and V has defending power $1/(2\beta)$. Therefore, the defending result of this fractional strategy is $1-1/(2\beta)$, which implies that $\mathsf{OPT}_f(R)\le 1-1/(2\beta)$. Next we show that for every $\mathbf{r}\in\Omega_p(\beta\cdot R)$, the number of well defended nodes is at most $2\beta\cdot R$. Suppose otherwise, there must exist a well defended node u with $r_u<\frac{\beta\cdot R}{2\beta\cdot R}=0.5$. Hence

$$\begin{aligned}\pi_u(\mathbf{r})&=r_u+\textstyle\sum_{v\in N(u)}w_{uv}\cdot r_v\\&\le r_u+\tfrac{1}{2\beta\cdot R}\cdot(\beta\cdot R-r_u)<\tfrac{1}{2}+\tfrac{1}{2}=1.\end{aligned}$$

However, since u is well defended, we must have $\pi_u(\mathbf{r})\ge 1$, which is a contradiction.

Hence for any pure strategy $\mathbf{r}\in\Omega_p(\beta\cdot R)$, we have $\|\mathbf{x}(\mathbf{r})\|\le 2\beta\cdot R$. Consequently for any mixed strategy $(D,\mathbf{p})\in\Omega_m(\beta\cdot R)$, we have $\|\mathbf{x}(D,\mathbf{p})\|\le 2\beta\cdot R$ (because $\mathbf{x}(D,\mathbf{p})$ is a linear combination of defending statuses of pure strategies). Hence there must exist a node u for which

$$x_u(D,\mathbf{p})\le\frac{2\beta\cdot R}{|U|+|V|}=\frac{2\beta\cdot R}{2\beta\cdot R+4\beta^2\cdot R}<\frac{1}{2\beta}.$$

Therefore, $\mathsf{OPT}_m(\beta\cdot R)>1-1/(2\beta)\ge\mathsf{OPT}_f(R)$. □

4. Small support mixed strategies

So far, we evaluate the quality of a mixed strategy only by its defending result without considering its support size $|D|$. Intuitively, the larger support a mixed strategy has, the more likely the strategy can balance the defending status among all nodes. However, in practice, it is usually preferable to have mixed strategies $(D,\mathbf{p})$ with a small D for efficiency purpose. In this section, we study the computation of mixed strategies that have good defending results and small support. In particular, we propose the Patching algorithm that computes mixed strategies with an upper bound on the support size, but also have good defending results. By Theorem 3.2, we know that computing the optimal mixed strategy is NP-hard. Moreover, by the reduction we can see that even computing the optimal mixed strategy with $|D|=2$ is NP-hard. However, we have the following very helpful observations. We show that deciding if a set of nodes can be defended simultaneously using one pure strategy is polynomial-time solvable. Throughout this section we fix $G(V,E)$ to be the graph instance and R to be the total resource.

Lemma 4.1. *Given a set of nodes $S\subseteq V$, deciding if there exists $\mathbf{r}\in\Omega_p$ with $x_u(\mathbf{r})=1$ for all $u\in S$ is polynomial-time solvable. Moreover, if they exist, we can compute one in polynomial time.*

Proof. We can reduce the problem of defending all nodes in S with one pure strategy (using total resource R) to solving the following feasibility LP. In particular, we introduce the variable r_u to denote the resource allocated to node u. We introduce the constraints that total resource used is at most R, and that each node $u\in S$ has defending power at least θ_u.

$$\begin{aligned}&\text{minimize}\quad && 0\\&\text{subject to}\quad && \textstyle\sum_{u\in V}r_u\le R,\\&&& r_u+\textstyle\sum_{v\in N(u)}w_{vu}\cdot r_v\ge\theta_u,\qquad \forall u\in S\end{aligned}$$

Algorithm 5: Patching.

Input: the number of iterations d and optimal pure strategy $\mathbf{r}^*$
$D \leftarrow \{\mathbf{r}^*\}$;
for $i = 2, 3, \ldots, d$ **do**
  $\mathbf{p} \leftarrow \mathsf{ProbLP}(D)$;
  compute the loss vector L_m of mixed strategy $(D, \mathbf{p})$;
  $\mathbf{r} \leftarrow \mathsf{FindR}(L_m)$;
  if $\|\mathbf{r}\| \neq 0$ **then**
    $D \leftarrow D \cup \{\mathbf{r}\}$;
$\mathbf{p} \leftarrow \mathsf{ProbLP}(D)$;
return $(D, \mathbf{p})$
Output: mixed strategy set $(D, \mathbf{p})$

$$r_u \geq 0, \qquad \forall u \in V.$$

If the above LP is infeasible then there does not exist a pure strategy that can defend all nodes in S; otherwise any feasible solution to the LP is the desired pure strategy. □

While computing the optimal mixed strategy is NP-hard, we show that for a small set of pure strategies D, computing the optimal mixed strategy with support D is polynomial-time solvable.

Lemma 4.2. *Given a set of pure strategies $D \subseteq \Omega_p$, the optimal mixed strategy $(D, \mathbf{p}^*)$ with support D can be computed in time polynomial in $|D|$ and n.*

Proof. Since D is fixed, the problem is to decide the probability $p(\mathbf{r})$ for each $\mathbf{r} \in D$, such that the defending result is as small as possible. We first compute the defending status $\mathbf{x}(\mathbf{r})$ for each $\mathbf{r} \in D$. Then we transform this problem into an LP, in which the probabilities $\{p(\mathbf{r})\}_{\mathbf{r}\in D}$ and L are variables.

$$\begin{aligned} \text{minimize} \quad & L \\ \text{subject to} \quad & \textstyle\sum_{\mathbf{r}\in D} p(\mathbf{r}) = 1, \\ & (1 - \textstyle\sum_{\mathbf{r}\in D} p(\mathbf{r}) \cdot x_u(\mathbf{r})) \cdot \alpha_u \leq L, \ \forall u \in V \\ & p(\mathbf{r}) \geq 0, \qquad \forall \mathbf{r} \in D. \end{aligned}$$

It can be verified that the optimal solution to the above LP corresponds to the mixed strategy with support D that has minimum loss. The first and third sets of constraints guarantee that $\{p(\mathbf{r})\}_{\mathbf{r}\in D}$ is a feasible probability distribution over D. The second set of constraints guarantees that the final defending result is minimum. □

4.1. The Patching algorithm

Following the above observations, we propose the local-search based algorithm that progressively and efficiently computes a mixed strategy with small support and good defending result. Our algorithm takes as input an iteration bound d, terminates after d search steps and outputs a mixed strategy $(D, \mathbf{p})$ with $|D| \leq d$. For convenience of notation we use $L_m(u)$ to denote $L_m(u, (D, \mathbf{p}))$, when the mixed strategy $(D, \mathbf{p})$ is clear from the context.

Intuitively speaking, our algorithm starts from a mixed strategy $(D, \mathbf{p})$ and tries to include a new pure strategy $\mathbf{r}$ into D, so that the optimal mixed strategy with support $D \cup \{\mathbf{r}\}$ is likely to achieve a better defending result. As shown in Lemma 4.2, as long as $|D|$ is small, computing the optimal mixed strategy with support D can be done efficiently by solving an LP. We denote this sub-routine by $\mathsf{ProbLP}(D)$. Our main idea is to add the new strategy to patch the poorly defended nodes up based on their current losses. Borrowing some ideas from the proof of Lemma 3.4, we compute the maximal set of nodes M with largest losses under the current mixed strategy $(D, \mathbf{p})$, and use Lemma 4.1 to compute a new pure strategy that defends these nodes. As we will show in the next section, as long as the maximum loss of nodes in M is larger than that of nodes not in M, our algorithm can always make progress in decreasing the defending result. Otherwise we randomly permute the nodes in V and try to include a random new pure strategy into D. We introduce the $\mathsf{FindR}(L_m)$ subroutine for the computation of the new pure strategy, for a given loss vector L_m. Note that if it fails to compute a new pure strategy, an all-zero vector will be returned. We summarize the main steps of the Patching algorithm in Algorithm 5. Initially we set the strategy set D to be a singleton containing only the optimal pure strategy and the algorithm terminates after d iterations.

Next we introduce the details of the sub-routine FindR. As discussed, given the loss vector L_m, the idea is to first locate the nodes with large losses and then generate a new pure strategy that enhances the defending statuses of these poorly defended nodes. We thus use similar ideas as in the proof of Lemma 3.4 to compute the maximal set of nodes to be defended. However, since resource sharing is considered, the procedure is slightly more complicated.

Algorithm 6: FindR.

Input: the loss vector L_m with current strategy set D
1 let $M \leftarrow \text{SharedMaxTop}(L_m)$;
2 **if** $\exists\, \mathbf{r} \in D : x_u(\mathbf{r}) = 1$ *for all* $u \in M$ **then**
3 $L_m \leftarrow$ random vector in $[0,1]^V$ and $M \leftarrow \text{SharedMaxTop}(L_m)$;
4 **if** $\exists\, \mathbf{r} \in D : x_u(\mathbf{r}) = 1$ *for all* $u \in M$ **then**
5 **return** $\{0\}^V$
6 let $\mathbf{r}$ be the pure strategy that defends all nodes in M ;
7 **return** $\mathbf{r}$
Output: new pure strategy $\mathbf{r}$

Given any integer $k > 0$, we can identify the Top-k nodes S with maximal losses in L_m, and check whether it is possible to defend all nodes in S by solving an LP (see Lemma 4.1). Using a binary search on k we can identify the maximum k for which the corresponding set of nodes S can be defended. Let $\text{SharedMaxTop}(L_m)$ be these nodes. The sub-routine $\text{FindR}(L_m)$ first computes $M \leftarrow \text{SharedMaxTop}(L_m)$ and tries to include the pure strategy $\mathbf{r}$ that defends all nodes in M. If there already exists a strategy in D that defends all nodes in M, then it is unnecessary to include $\mathbf{r}$ because its inclusion will not help in decreasing the defending result. In such case we do a random permutation on V (by replacing L_m with a random vector in $[0,1]^V$), and compute another pure strategy. As mentioned, if we fail to find a new pure strategy after the random permutation, then the sub-routine returns the trivial defending strategy $\{0\}^V$. We summarize the steps of $\text{FindR}(L_m)$ in Algorithm 6.

Complexity. Observe that every call to SharedMaxTop involves $O(\log n)$ computations of some feasibility LPs with $O(n)$ variables. Therefore, the total complexity of the FindR algorithm is bounded by $O(\log n)$ computations of LP solvings. Note that the complexity of each iteration of the Patching algorithm is dominated by the FindR sub-routine (recall that ProbLP can be done by solving one LP). As a consequence, the total complexity of Patching is bounded by $O(d \log n)$ computations of LP solvings, where d is the number of iterations.

4.2. Effectiveness

As we will show in our experiments (Section 6), the Patching algorithm achieves close-to-optimal defending results on several large datasets. In this section, we theoretically analyze the algorithm and formalize the condition under which our algorithm is guaranteed to make progress in terms of decreasing the defending result. Our analysis also sheds lights into why random permutation could help improve the performance of the algorithm.

Consider any iteration of the Patching algorithm. Suppose $(D, \mathbf{p})$ is the current mixed strategy and L_m is the loss vector. In line 5 of Algorithm 5, we call the sub-routine FindR. In the sub-routine we compute the maximal set of nodes that can be defended $M \leftarrow \text{SharedMaxTop}(L_m)$ (line 1 of Algorithm 6). The following lemma states that as long as M contains all nodes with maximum loss (in which case $\Delta L > 0$), our algorithm can always make progress in decreasing the defending result.

Lemma 4.3. *Let* $\mathbf{r}$ *be the pure strategy that defends all nodes in* M*. Including* $\mathbf{r}$ *into* D *decreases the defending result of the current mixed strategy* $(D, \mathbf{p})$ *by at least* $\frac{\Delta L}{\Delta L + \alpha_{max}} \cdot L_m(D, \mathbf{p})$*, where*

$$\alpha_{max} = \max_{u \in V}\{\alpha_u\} \quad \textit{and} \quad \Delta L = \max_{u \in M}\{L_m(u)\} - \max_{u \notin M}\{L_m(u)\}.$$

Proof. Let $\epsilon = \frac{\Delta L}{\Delta L + \alpha_{\max}}$ and $D' = D \cup \{\mathbf{r}\}$. We show that there exists a mixed strategy with support D' that achieves defending result $(1-\epsilon) \cdot L_m(D, \mathbf{p})$. Specifically, we define the mixed strategy $(D', \mathbf{p}')$ as follows. Let $p'(\mathbf{r}) = \epsilon$; for each $\mathbf{r}' \in D$, let $p'(\mathbf{r}') = (1-\epsilon) \cdot p(\mathbf{r}')$. Since $\|\mathbf{p}\| = 1$, we have

$$\|\mathbf{p}'\| = \epsilon + (1-\epsilon) \cdot \|\mathbf{p}\| = 1.$$

Thus $(D', \mathbf{p}')$ is a feasible mixed strategy. For each $v \notin M$, since v is not defended by the new strategy $\mathbf{r}$, we have

$$\begin{aligned} &L_m(v, (D', \mathbf{p}')) = \left(1 - \textstyle\sum_{\mathbf{r} \in D'} p'(\mathbf{r}) \cdot x_v(\mathbf{r})\right) \cdot \alpha_v \\ =& \left(1 - \textstyle\sum_{\mathbf{r} \in D} p'(\mathbf{r}) \cdot x_v(\mathbf{r})\right) \cdot \alpha_v \\ =& \left(1 - (1-\epsilon) \cdot \textstyle\sum_{\mathbf{r} \in D} p(\mathbf{r}) \cdot x_v(\mathbf{r})\right) \cdot \alpha_v \\ =& (1-\epsilon) \cdot L_m(v, (D, \mathbf{p})) + \epsilon \cdot \alpha_v \\ \leq& (1-\epsilon) \cdot (L_m(D, \mathbf{p}) - \Delta L) + \epsilon \cdot \alpha_{\max} = (1-\epsilon) \cdot L_m(D, \mathbf{p}). \end{aligned}$$

For each $u \in M$, we have

$$L_m(u, (D', \mathbf{p}')) = \left(1 - \textstyle\sum_{\mathbf{r} \in D'} p'(\mathbf{r}) \cdot x_u(\mathbf{r})\right) \cdot \alpha_u$$

$$= \left(1 - (1-\epsilon) \cdot \sum_{\mathbf{r} \in D} p(\mathbf{r}) \cdot x_u(\mathbf{r}) - \epsilon \cdot 1\right) \cdot \alpha_u$$
$$= (1-\epsilon) \cdot L_m(u, (D, \mathbf{p})) \leq (1-\epsilon) \cdot L_m(D, \mathbf{p}).$$

Hence we have $L_m(D', \mathbf{p}') \leq (1-\epsilon) \cdot L_m(D, \mathbf{p})$. So given $D' = D \cup \{\mathbf{r}\}$, when our algorithm computes the optimal mixed strategy with support D', its defending result must be at most $(1-\epsilon) \cdot L_m(D, \mathbf{p})$, as claimed by the lemma. □

By Lemma 4.3, we can see that as long as all nodes with maximum loss can be defended by one pure strategy, the Patching algorithm can always decrease the defending result. When the nodes with maximum loss are too many and no pure strategy can defend them all, we randomly permute the nodes to compute a random pure strategy to be included in D. As we observed from our empirical study, such random permutations are crucial as otherwise the algorithm may get stuck in the early stage during the execution.

5. Uniform attacks

In contrast to the adversarial attack scenarios discussed in the preceding section—where the attacker targets a specific node to maximize the defender's loss—this section explores the case of uniform attacks. Note that the adversarial attack model and the uniform attack model correspond to the cases when the attacker is fully rational and fully irrational, respectively. Under uniform attacks, each node is equally likely to be attacked, regardless of its defense status. Next, we introduce definitions and notations for the uniform attack setting. The network is again represented as an undirected connected graph $G(V, E)$ with n nodes.

Definition 5.1 *(Defending result of uniform attack).* Given a strategy $\mathbf{r}$, the defending power of node u is $\pi_u(\mathbf{r}) = r_u + \sum_{v \in N(u)} w_{uv} \cdot r_v$. For a pure strategy, the defending status $x_u(\mathbf{r})$ of u is the indicator of whether $\pi_u(\mathbf{r}) \geq \theta_u$; for a fractional strategy, the defending status $\tilde{x}_u(\mathbf{r}) = \min\{\frac{\pi_u(\mathbf{r})}{\theta_u}, 1\}$. For a pure strategy, the loss at a node u is defined as $L_p(u, \mathbf{r}) = \alpha_u \cdot (1 - x_u(\mathbf{r}))$. The overall defending result is then expressed as:

$$L_p^{\text{uni}}(\mathbf{r}) = \frac{1}{n} \sum_{u \in V} L_p(u, \mathbf{r}).$$

The losses under fractional strategy $\mathbf{r}$ are defined similarly: $L_f(u, \mathbf{r}) = \alpha_u \cdot (1 - \tilde{x}_u(\mathbf{r}))$ and $L_f^{\text{uni}}(\mathbf{r}) = \frac{1}{n} \sum_{u \in V} L_f(u, \mathbf{r})$.

To illustrate the distinction between adversarial and uniform attacks, consider the following example:

Example 5.2. Echoing Example 2.9, consider three targets with values and thresholds listed in the table below. With total resource $R = 5$, an optimal pure strategy is $(4, 0, 1)$, resulting in a defending outcome of $\mathsf{OPT}_p = 2$ under an adaptive attack. However, under a uniform attack, the expected loss diminishes to $\frac{2}{3}$, as node b will be attacked only with probability $1/3$. Note that for this example the optimal fractional strategy is $(1, 3, 1)$, with $\mathsf{OPT}_f = \frac{1}{2}$.

Target	a	b	c
Value	2	2	1
Threshold	4	3	1

As in Section 2, we use $\mathsf{OPT}_p^{\text{uni}}$, $\mathsf{OPT}_m^{\text{uni}}$, and $\mathsf{OPT}_f^{\text{uni}}$ to represent the defending results of the optimal pure, mixed, and fractional strategies under uniform attacks. The other notations are defined in a similar way. As the example above shows, the optimal defending results by pure and fractional strategies might be different. However, we show that for uniform attacks, the optimal mixed strategy is a pure strategy.

5.1. Equivalence and hardness of mixed and pure strategies

Unlike the case of adversarial attacks, we show that for uniform attacks, the mixed strategies do not have advantage over pure strategies.

Lemma 5.3. *For uniform attacks, every mixed strategy $(D, \mathbf{p})$ can be transformed into a pure strategy, whose defending result is at most that of the mixed strategy.*

Proof. Recall that the loss of a node u under a mixed strategy $(D, \mathbf{p})$ is the expected loss of the node under the distribution $\mathbf{p}$ over pure strategies D:

$$L_m(u,(D,\mathbf{p})) = \alpha_u \cdot (1 - x_u(D,\mathbf{p})) \\ = \alpha_u \cdot (1 - \sum_{\mathbf{r}\in D} p(\mathbf{r}) \cdot x_u(\mathbf{r})) = \sum_{\mathbf{r}\in D} p(\mathbf{r}) \cdot L_p(u,\mathbf{r}).$$

By linearity of expectation we also have

$$L_m^{\text{uni}}(D,\mathbf{p}) = \frac{1}{n}\sum_{u\in V} L_m(u,(D,\mathbf{p})) = \sum_{\mathbf{r}\in D} p(\mathbf{r}) \cdot L_p^{\text{uni}}(\mathbf{r}).$$

Therefore, there must exist a pure strategy in D whose defending result is at most that of $(D,\mathbf{p})$, which concludes the proof. □

Since every pure strategy can be interpreted as a mixed strategy with $|D| = 1$, the above lemma implies that the optimal mixed and pure strategies are equivalent. Thus in the following we only study pure strategies and fractional strategies. We first show that computing the optimal pure strategy is NP-hard, even in the isolated model or the uniform threshold model.

Theorem 5.4. *Unless $P = NP$, there does not exist any polynomial-time algorithm that, given a graph $G(V,E)$ and resource R, computes the optimal pure strategy, even under the isolated model or the uniform threshold model.*

Proof. We first show the hardness result under the isolated model. We prove the hardness result through a reduction from the Knapsack problem. In an instance of the knapsack problem, we have a capacity C and a set of items $\{e_1, e_2, \dots, e_n\}$, where each item e_i has value v_i and size s_i. The goal is to find a subset of items whose total size is at most the capacity C while its total value is maximized. Given the instance we construct an instance $G(V,E)$ under the isolated model as follows. Let $V = \{u_1, u_2, \dots, u_n\}$, where each node u_i has value $\alpha_{u_i} = v_i$ and threshold $\theta_{u_i} = s_i$. Let the capacity $C = R$. The defending result of any pure strategy $\mathbf{r}$ is given by

$$L_p^{\text{uni}}(\mathbf{r}) = \frac{1}{n}\sum_{u\in V} L_p(u,\mathbf{r}) = \frac{1}{n}\cdot\left(\sum_{u\in V}\alpha_u - \sum_{u\in V: r_u \ge \theta_u}\alpha_u\right).$$

Therefore, minimizing the defending result is equivalent to find a subset of nodes whose total threshold is at most R, while its total value is maximized. Hence there is a one-one correspondence between a pure strategy and a solution to the knapsack problem. Since the reduction can be done in polynomial time, we conclude that the computation of optimal pure strategy is NP-hard, even in the isolated model.

Next we prove that the problem is NP-hard under the uniform threshold model, when resource sharing is enabled. We prove the result by a reduction from the Maximum Coverage problem, in which we have a set of elements $U = \{e_1, e_2, \dots, e_n\}$ and a collection of sets $\{S_1, S_2, \dots, S_m\}$, where each set $S_i \subseteq U$ covers some elements. The goal is to pick k sets, where k is a parameter of the problem, that covers a maximum number of elements.

Given any instance of the maximum coverage problem, we construct a uniform threshold instance $G(V,E)$ with $n+m$ nodes, where the first n nodes represent the n elements and the last m nodes represent the m sets. Specifically, let $V = \{u_1, u_2, \dots, u_{n+m}\}$, where $\alpha_{u_i} = 1$ for all $i \le n$ and $\alpha_{u_i} = 0$ for all $i > n$. Let there be an edge between u_i and u_{n+j} with weight 1 if e_i is contained in set S_j (in the maximum coverage problem). We call u_i an *element-node* if $i \le n$, *set-node* if $i > n$. Finally, let $R = k$.

Clearly, the optimal pure strategy for the problem will ensure that a maximum number of element-nodes are well defended. Therefore, it is optimal to only place defending resources on the set-nodes, since one unit of resource on node u_{n+j} protects all elements covered by S_j. Hence computing the optimal pure strategy corresponds to finding k sets that covers most elements. Since the reduction can be done in polynomial time, the result follows. □

5.2. Pure and fractional strategies in the isolated model

As we have shown above, for the isolated model, the problem of computing the optimal pure strategy is equivalent to the knapsack problem. Therefore when the defending thresholds of all nodes are integers, the optimal pure strategy can be efficiently computed using dynamic programming, yielding a complexity of $O(nR)$. For general (non-integer) thresholds, the problem can be solved by the following Mixed Integer Linear Program (MILP)[5]:

$$\begin{aligned}
(\mathsf{LP}_f^{\text{uni}}(R)) \quad & \text{minimize} && L \\
& \text{subject to} && \textstyle\sum_{u\in V} r_u \le R, \\
& && r_u \ge \theta_u \cdot x_u, \quad \forall u \in V \\
& && \tfrac{1}{n}\cdot\textstyle\sum_{u\in V}(1-x_u)\cdot\alpha_u \le L \\
& && x_u \in \{0,1\}, \quad \forall u \in V
\end{aligned}$$

[5] By changing the second set of constraints to $r_u + \sum_{v\in N(u)} w_{uv}\cdot r_v \ge \theta_u \cdot x_u$, $\forall u \in V$, we can use the MILP to solve the non-isolated model.

Here, for each node $u \in V$, we introduce an integer variable $x_u \in \{0,1\}$ indicating whether $r_u \geq \theta_u$, which is exactly the defending status of the node under the pure strategy. This MILP facilitates the computation of the optimal pure strategy and its corresponding value $\mathsf{OPT}_p^{\text{uni}}(R)$.

For the fractional strategy in the uniform attack setting, we relax the integer constraint on x_u to allow continuous values ($x_u \in [0,1]$), where x_u represents the defending status of u under the fractional strategy. Therefore by solving the LP (in polynomial time), the optimal solution gives an optimal fractional strategy, with an objective equal to $\mathsf{OPT}_f^{\text{uni}}(R)$.

In fact, by the connection we have established in the proof of Theorem 5.4 between the defending problem and the knapsack problem, the computation of fractional strategy is equivalent to the fractional knapsack problem, in which an item can be selected fractionally. For the fractional knapsack problem, it is well known that the greedy algorithm that selects items continuously in descending order of density (v_i/s_i) until the total (fractional) size reaches R gives an optimal solution. Therefore, the optimal fractional strategy can be computed in $O(n \log n)$ time. Since in the optimal fractional defending strategy, the decrease in defending result per unit of defending resource is diminishing, the defending result is a convex function of the total resource R. Thus we have the following lemma immediately (we can also prove the lemma using a proof similar to that of Lemma 3.1).

Lemma 5.5 *(Convexity). Given resource R_1 and R_2, we have*

$$OPT_f^{uni}(R_1) + OPT_f^{uni}(R_2) \geq 2 \cdot OPT_f^{uni}\left(\tfrac{1}{2}(R_1 + R_2)\right).$$

While it is NP-hard to compute the optimal pure strategy, we can use $\mathsf{OPT}_f^{\text{uni}}(R)$ to provide a lower bound on $\mathsf{OPT}_p^{\text{uni}}(R)$. We further show that we can give an upper bound for $\mathsf{OPT}_p^{\text{uni}}(R)$ similar to Theorem 3.3, echoing the result for adversarial attack from Section 3.3.

Theorem 5.6. *In the isolated model, for any instance $G(V,E)$ and total resource R, we have*

$$OPT_p^{uni}(R) \leq OPT_f^{uni}(R - \theta_{max}),$$

where $\theta_{max} = \max_{u \in V}\{\theta_u\}$ is the maximum threshold among the nodes.

Proof. As we have argued above, the optimal fractional strategy can be computed by first sorting all nodes in descending order of density α_u/θ_u, and then allocate resources continuously in the sorted order until the resource is used up. Therefore, among the nodes that receive resources, at most one of them (e.g., the last node) is fractionally defended. Let S be the set of nodes that are fully defended, i.e., those with $r_u = \theta_u$. The total resource allocated to nodes in S is larger than $R - \theta_{\max}$ but at most R. Hence we can define a pure strategy $\mathbf{r}$ with resource R that defends all nodes in S, which implies

$$\mathsf{OPT}_p^{\text{uni}}(R) \leq L_p^{\text{uni}}(\mathbf{r}) = \mathsf{OPT}_f^{\text{uni}}\Big(\sum_{u \in S} \theta_u\Big) \leq \mathsf{OPT}_f^{\text{uni}}(R - \theta_{\max}). \quad \square$$

By the convexity of the function $\mathsf{OPT}_f^{\text{uni}}(\cdot)$, we also have

$$\mathsf{OPT}_f^{\text{uni}}(R - \theta_{\max}) \leq \mathsf{OPT}_f^{\text{uni}}(R) + \frac{\theta_{\max}}{R} \cdot (\max_{u \in V}\{\alpha_u\} - \mathsf{OPT}_f^{\text{uni}}(R)),$$

which implies that when $R \gg \theta_{\max}$, the values of $\mathsf{OPT}_f(R)$ and $\mathsf{OPT}_f(R - \theta_{\max})$ are very close.

6. Experimental evaluation

In this section we perform the experimental evaluation of our algorithms on several real-world graph datasets whose sizes range from 1000 nodes to 260k nodes (see Table 2). All the datasets are downloaded from SNAP by Stanford [38]. Unless otherwise specified, we set the parameters of instances as follows.[6] For each of these datasets, we set the value α_u of each node u to be an independent random integer chosen uniformly at random from $[1,9]$. We set the threshold θ_u of each node u differently for different sets of experiments. We set the weight w_{uv} of each edge $(u,v) \in E$ to be an independent random real number chosen from $[0,1]$. We set the total resource $R = 0.2 \cdot \sum_{u \in V} \theta_u$ by default, and will evaluate the results for different settings of total resource.

In the experiments we mainly evaluate the effectiveness of the Patching algorithm. Additionally we test and report the mixed strategies we stated in Section 3.3, and compare their defending results and support sizes with that of the mixed strategies returned by Patching. As we have shown in Section 3, for the same problem instance we always have $\mathsf{OPT}_m \geq \mathsf{OPT}_f$. Thus in our experiments we mainly use OPT_f as the baseline to evaluate the performance of the mixed strategies.

Experiment environment. We perform our experiments on an AWS Ubuntu 18.04 machine with 32 threads and 128 GB RAM without GPU. We use Gurobi optimizer as our solver for the LPs.

[6] We remark that for different settings of the parameters, e.g., wider ranges for the values and thresholds, or smaller values of R, the experimental results are very similar.

Table 2
Number of nodes and edges of the datasets.

Dataset	Email-S	Facebook	Ca-AstroPh	Email-L	Twitter	Amazon
# Nodes	1,005	4,039	18,772	36,692	81,306	262,111
# Edges	25,571	88,234	198,110	367,662	1,768,149	1,234,877

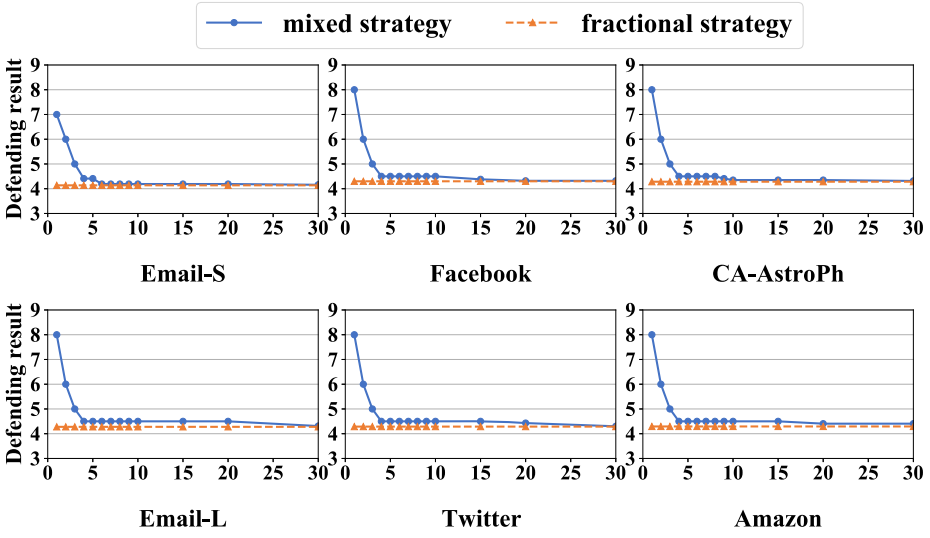


Fig. 1. Defending Results of Mixed Strategies by Patching in the Uniform Threshold Isolated Model when $R = 0.2 \cdot \sum_{u \in V} \theta_u$.

6.1. Adversarial attack: uniform threshold in the isolated model

We first consider the most basic setting with uniform thresholds and without resource sharing. The same setting was also considered in [11,12]. That is, we set $\theta_u = 1$ for all nodes $u \in V$ and $w_{uv} = 0$ for all edges $(u, v) \in E$, in this subsection. Under this setting it can be shown that $\mathsf{OPT}_m = \mathsf{OPT}_f$ for all instances of the problem [11,12]. Thus we measure the effectiveness of the Patching algorithm by comparing the defending result of the returned mixed strategy with OPT_f. For each dataset, we report the defending result of the mixed strategy returned by Patching(d), for increasing values of $d \in [1, 30]$. Note that for $d = 1$, the mixed strategy uses the optimal pure strategy $\mathbf{r}^*$ with probability 1. For general d, the returned mixed strategy uses $|D| \le d$ pure strategies. The results are presented as Fig. 1.

From Fig. 1, we observe that the defending result rapidly decreases in the first few iterations of the algorithm, and gradually converges to the optimal defending result OPT_f. Specifically, within the first 5 iterations, the defending result of the mixed strategy is already within a 5% difference with the optimal one in most datasets. After 30 iterations, the defending result is almost identical to the optimal one in all datasets.

We also evaluate the results under different total resource, by setting $R = 0.1 \cdot \sum_{u \in V} \theta_u$. From Fig. 2, we observe a similar phenomenon as in Fig. 1: the defending result decreases rapidly within the first few iterations, and then gradually converge to that of the optimal fractional strategy. After 120 iterations, the defending result is almost identical to the optimal one in all datasets. As expected, when the amount of total resource is smaller, the support of the mixed strategy will increase: recall that when $V_{\max}(\mathbf{f}) \subsetneq \mathrm{MaxTop}(\mathbf{f})$ (which is more likely to happen when R is small), we need to do a random permutation on the nodes to find a new pure strategy. Therefore the algorithm takes more iterations to eventually converge.

The experiment demonstrates the effectiveness of our algorithm on computing mixed strategies that have small support sizes and are close-to-optimal.

6.2. Adversarial attack: general thresholds in the isolated model

Next we consider the more general setting with non-uniform thresholds, in which the threshold $\theta_u \in [1, 10]$ is chosen uniformly at random for each node $u \in V$. As we have shown in Theorem 3.2, computing the optimal mixed strategy in this case is NP-hard. On the other hand, we have shown in Section 3.3 that for any instance we always have $\mathsf{OPT}_f(R) \le \mathsf{OPT}_m(R) \le \mathsf{OPT}_f(R - \theta_{\max})$, where the maximum threshold $\theta_{\max} \le 10$ in our experiments.

The experimental results are reported in Fig. 3 and Fig. 4, for $R = 0.2 \cdot \sum_{u \in V} \theta_u$ and $R = 0.1 \cdot \sum_{u \in V} \theta_u$, respectively. As we can observe, the results are very similar to the uniform threshold case we have considered in the previous experiments: the defending results of the algorithm quickly converge to that of the optimal fractional strategy, where the convergence is faster when R is larger. Nevertheless, both experiments confirm our theoretical analysis in Section 3.3: when R is sufficiently large (compared with $\theta_{\max}$),

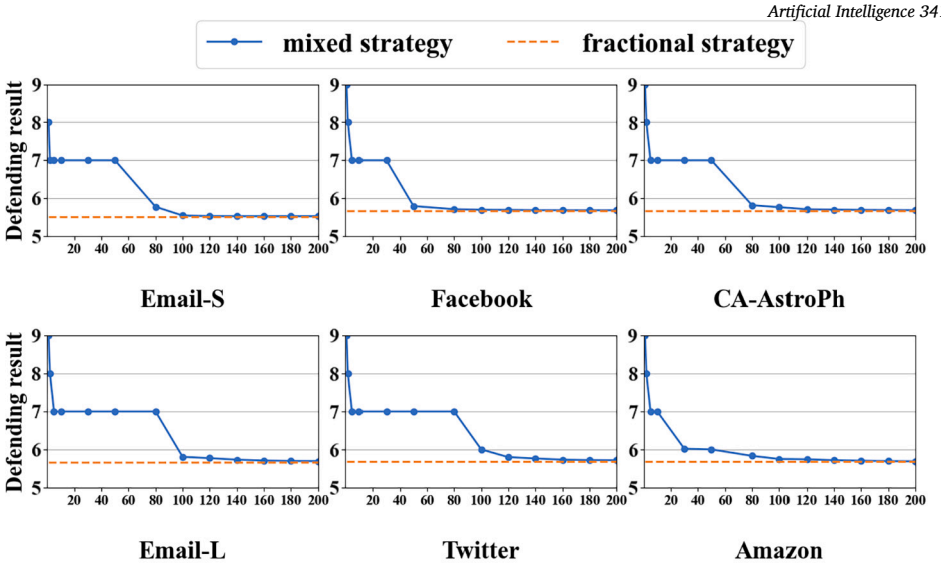


Fig. 2. Defending Results of Mixed Strategies by Patching in Uniform Threshold Isolated Model when $R = 0.1 \cdot \sum_{u \in V} \theta_u$.

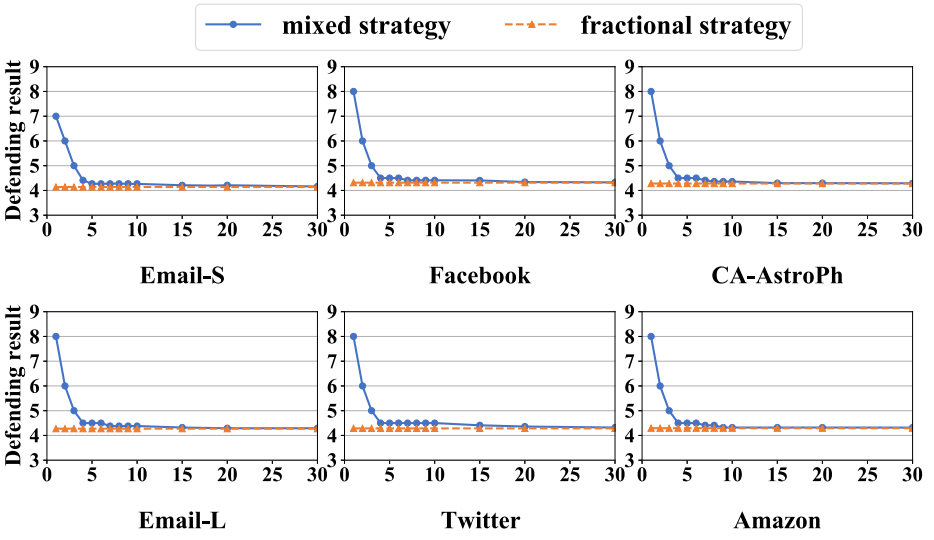


Fig. 3. Defending Results of Mixed Strategies by Patching in General Threshold Isolated Model when $R = 0.2 \cdot \sum_{u \in V} \theta_u$.

the optimal mixed strategy should have defending result close to $\mathsf{OPT}_f(R)$. Furthermore, the experiment demonstrates that even for general thresholds, Patching returns mixed strategies that are close-to-optimal, and uses very few pure strategies.

Recall that in Section 3.3 we show that there exists a mixed strategy $(D, \mathbf{p}) \in \Omega_m(R)$ whose defending result $L_m(D, \mathbf{p}) = \mathsf{OPT}_f(R - \theta_{\max})$. In our experiments, we implement the algorithm and report the defending result and the support size to verify its correctness. The results are presented in Table 3, where $(D, \mathbf{p})$ is the mixed strategy our algorithm computes, and $(D, \mathbf{p}^*)$ is the optimal mixed strategy with support D (which can be computed by solving an LP, as we have shown in Lemma 4.2). Note that due to long computation time, we did not finish the computing of $L_m(D, \mathbf{p}^*)$ for the Twitter and Amazon datasets (too many variables). In the table, we also compare them with the mixed strategies returned by Patching(d) with $d \in \{5, 30\}$. Consistent with our theoretical analysis, for all datasets we have

$$\mathsf{OPT}_f(R - \theta_{\max}) = L_m(D, \mathbf{p}) \geq L_m(D, \mathbf{p}^*) \geq \mathsf{OPT}_f(R).$$

From the above results we can see that compared to $(D, \mathbf{p})$, the Patching algorithm is able to compute mixed strategies with very small support, e.g., 30 vs. 2500+ for large datasets, while guaranteeing a defending result that is very close.

We also compare the time to compute $L_m(D, \mathbf{p})$ and the running time (in seconds) of Patching in Table 4. As we can observe from the table, compared to the computation of $L_m(D, \mathbf{p})$, the running time of Patching is less sensitive to the size of the network. Consequently for large datasets the Patching algorithm runs several times faster than computing $L_m(D, \mathbf{p})$. In conclusion, the Patching

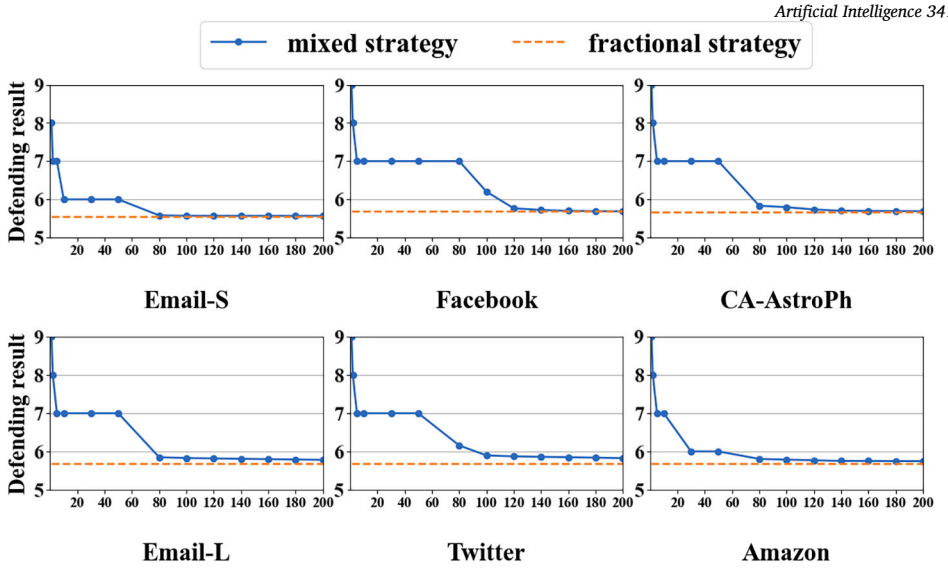


Fig. 4. Defending Results of Mixed Strategies by Patching in General Threshold Isolated Model when $R = 0.1 \cdot \sum_{u \in V} \theta_u$.

Table 3
Defending Results of Mixed Strategies in Different Datasets.

	Email-S	Facebook	Ca-AstroPh	Email-L	Twitter	Amazon
$\mathsf{OPT}_f(R - \theta_{\max})$	4.161	4.32	4.281	4.273	4.285	4.293
$L_m(D, \mathbf{p})$	4.161	4.32	4.281	4.273	4.285	4.293
$L_m(D, \mathbf{p}^*)$	4.147	4.316	4.28	4.273	–	–
$\lvert D \rvert$	268	671	1900	2592	6052	9957
$\mathsf{OPT}_f(R)$	4.139	4.314	4.28	4.273	4.285	4.293
Patching(5)	4.41	4.5	4.5	4.5	4.5	4.5
Patching(30)	4.161	4.326	4.29	4.291	4.324	4.319

Table 4
Running Time Comparison (in seconds).

	Email-S	Facebook	Ca-AstroPh	Email-L	Twitter	Amazon
$L_m(D, \mathbf{p})$	0.2	3.2	57	214	1027	9370
Patching(5)	0.3	0.8	3.3	6.5	14	45
Patching(30)	2.3	8.2	37	73	165	523

algorithm computes mixed strategies that use way fewer pure strategies than $(D, \mathbf{p})$ while having similar defending results. Furthermore, its running time is also much smaller in large datasets.

6.3. Adversarial attack: general thresholds with resource sharing

Finally, we evaluate the performance of the Patching algorithm on the network defending problem with resource sharing. In contrast to the isolated model, with resource sharing we can no longer guarantee $\mathsf{OPT}_m \approx \mathsf{OPT}_f$, even if R is sufficiently large (see Section 3.5 for the hard instance). In other words, the lower bound OPT_f we compare our mixed strategy with can possibly be much smaller than the optimal defending result OPT_m of mixed strategies. Moreover, with resource sharing we must set R to be smaller compared with previous experiments. If we set $R = 0.2 \cdot \sum_{u \in V} \theta_u$ as before, the defending result is 0, because when resource can be shared, it requires a much smaller amount of resource to fully defend all nodes. For this reason, we set $R = 0.1 \cdot \sum_{u \in V} \theta_u$ for the Email-EU and CA-AstroPh datasets, and $R = 0.015 \cdot \sum_{u \in V} \theta_u$ for the Facebook dataset, because slightly larger values of R will result in $\mathsf{OPT}_p(R) = 0$. The experimental results are reported as Fig. 5. As discussed in Section 4, computing mixed strategies for the non-isolated model involves solving LPs with $\Theta(n)$ variables, which can be quite time consuming. Thus we only manage to run the experiments on the three small datasets.

From Fig. 5, we observe similar phenomenons as in the isolated model: the defending result decreases dramatically in the first 5 iterations, and after around 10 iterations the defending result is close to what it will eventually converge to. However, different from the isolated model, now we can no longer guarantee that the defending results of the mixed strategies are close to OPT_f, the lower bound for OPT_m. As discussed above, one possible reason can be that OPT_f is much smaller than OPT_m. Unfortunately, unless

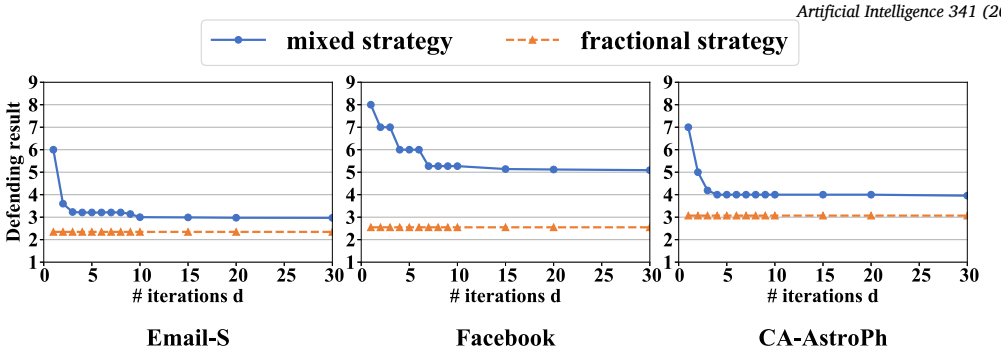


Fig. 5. Patching in the Non-isolated Model.

Table 5
Defending Results under Uniform Attack with Uniform Thresholds.

Dataset	Email-S	Facebook	Ca-AstroPh	Email-L	Twitter	Amazon
$\mathsf{OPT}_f^{\text{uni}}(R)$	3.1363	3.306	3.272	3.2742	3.28254	3.28745
$\mathsf{OPT}_p^{\text{uni}}(R)$	3.1363	3.3077	3.2722	3.2743	3.28256	3.28746
$\mathsf{OPT}_f^{\text{uni}}(R-\theta_{\max})$	3.143	3.308	3.2724	3.2744	3.28264	3.28748

Table 6
Defending Results under Uniform Attack with General Thresholds.

Dataset	Email-S	Facebook	Ca-AstroPh	Email-L	Twitter	Amazon
$\mathsf{OPT}_f^{\text{uni}}(R)$	2.488	2.6257	2.6078	2.6084	2.6144	2.6122
$\mathsf{OPT}_p^{\text{uni}}(R)$	2.489	2.6259	2.6079	2.6085	2.6144	2.6122
$\mathsf{OPT}_f^{\text{uni}}(R-\theta_{\max})$	2.5	2.6287	2.6085	2.6088	2.6145	2.6123

there is way to give a tighter lower bound for OPT_m, there is no way to find out whether the mixed strategy Patching(30) returns is close-to-optimal or not. We believe that this would be an interesting topic to study, and we leave it as the future work.

6.4. Uniform attack

In this section, we evaluate the defending results of different strategies under the uniform attack setting, verifying the theoretical results.

We start by considering the most basic setting with uniform thresholds in the isolated model. The parameter settings are the same as in Section 6.1. Recall from Lemma 5.3 that for uniform attacks the defending result of any mixed strategy is at least that of a pure strategy. Therefore we only evaluate the defending results of the optimal pure and fractional strategies, both of which can be computed via a reduction to the knapsack problem, as we have argued in Section 5. Given resource R, we compare $\mathsf{OPT}_p^{\text{uni}}(R)$ with $\mathsf{OPT}_f^{\text{uni}}(R)$ and $\mathsf{OPT}_f^{\text{uni}}(R-\theta_{\max})$. The results are shown in Table 5.

From Table 5, we observe that the defending result of the optimal pure strategy is almost equal to that of the optimal fractional strategy[7] and is less than $\mathsf{OPT}_f^{\text{uni}}(R-\theta_{\max})$ in all datasets. This demonstrates that $\mathsf{OPT}_f^{\text{uni}}(R-\theta_{\max})$ is an upper bound of $\mathsf{OPT}_p^{\text{uni}}(R)$, verifying the correctness of Lemma 5.6.

We also consider the setting with non-uniform thresholds, where θ_u is chosen uniformly at random from $[1, 10]$ for each node $u \in V$. As observed from Table 6, the results are very similar to the case of uniform thresholds.

Finally, we compare $\mathsf{OPT}_f^{\text{uni}}(R)$ and $\mathsf{OPT}_p^{\text{uni}}(R)$ in the general threshold setting when resource sharing is allowed. Unlike the isolated model, with resource sharing, we can no longer guarantee $\mathsf{OPT}_p^{\text{uni}}(R) \le \mathsf{OPT}_f^{\text{uni}}(R-\theta_{\max})$. Furthermore, since computing the optimal pure strategy for the non-isolated model involves solving an MILP with $\Theta(n)$ integer variables, which is time-consuming, we only ran experiments on the three small datasets. The results are shown in Table 7 (the parameter settings are the same as in Section 6.3).

From Table 7, we observe that there is a noticeable gap between $\mathsf{OPT}_p^{\text{uni}}(R)$ and $\mathsf{OPT}_f^{\text{uni}}(R)$, which shows that there is an integrality gap between the integral and fractional defending status. As we have shown in the proof of Theorem 5.4, when resource can be shared between neighboring nodes, the problem has a connection with the maximum coverage problem, which is not only NP-hard, but also admit a gap in the optimal integral solution and optimal fractional solution. Therefore, the experimental results are again consistent with our theoretical analysis.

[7] If R is divisible by $\theta_{\max}$, then the two defending results are equal.

Table 7
Defending Results in the Non-isolated Model.

Dataset	Email-S	Facebook	Ca-AstroPh
$\mathsf{OPT}_f^{\text{uni}}(R)$	0.3578	0.046	0.71
$\mathsf{OPT}_p^{\text{uni}}(R)$	0.403	0.11	1.22

7. Conclusion and future works

In this work, we study mixed strategies for security games with general threshold and quantify its advantage against pure strategies under adversarial attack and uniform attack. For both attack types, we show that it is NP-hard to compute the optimal mixed strategy in general and provide strong upper and lower bounds for the optimal defending result of mixed strategies for the isolated model. For adversarial attack, we propose the Patching algorithm for the computation of mixed strategies with theoretical guarantees. For uniform attack, we prove that the defending result of the optimal mixed strategy is equal to that of the optimal pure strategy, whose computation is NP-hard, even under the isolated model or the uniform threshold model.

Regarding future work, we believe that it would be most interesting to study mixed strategies against contagious attacks [19] and imperfect attackers [39]. The uniform attack setting and adversarial attack setting are two extreme cases of the quantal response model, i.e., the cases when the attacker is fully rational and fully irrational, respectively. Therefore, generalizing our results to quantal response model would be a very interesting future topic. Finally, since many real-world security games are not zero-sum, the problem of computing the mixed strategies with general defending requirement for non-zero-sum games is also worthwhile studying.

CRediT authorship contribution statement

Rufan Bai: Writing – original draft, Validation, Project administration, Methodology, Investigation, Formal analysis, Conceptualization. **Haoxing Lin:** Writing – review & editing, Data curation. **Xiaowei Wu:** Writing – original draft, Supervision, Methodology. **Minming Li:** Writing – review & editing. **Weijia Jia:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This research was funded by the National Natural Science Foundation of China (62402103) and the Natural Science Foundation of Jiangsu Province (BK20241273). This project was partially funded by the National Natural Science Foundation of China (52102400, 62203123), the Sustainable Development Science and Technology Project of Shenzhen Science and Technology Innovation Commission KCXFZ2020122-1173411032. Xiaowei Wu is funded by the Science and Technology Development Fund (FDCT), Macau SAR (file no. 0014/2022/AFJ, 0085/2022/A, and SKL-IOTSC-2024-2026). Weijia Jia is partially funded by the Chinese National Research Fund (NSFC) (62272050), and Zhuhai Science-Tech Innovation Bureau (2320004002772), and in part by the Interdisciplinary Intelligence SuperComputer Center of Beijing Normal University (Zhuhai).

Data availability

Data will be made available on request.

References

[1] E. Shieh, B. An, R. Yang, M. Tambe, C. Baldwin, J. DiRenzo, B. Maule, G. Meyer, PROTECT: a deployed game theoretic system to protect the ports of the United States, in: AAMAS, IFAAMAS, 2012, pp. 13–20.
[2] W. Guo, X. Wu, L. Wang, X. Xing, D. Song, PATROL: provable defense against adversarial policy in two-player games, in: USENIX Security Symposium, USENIX Association, 2023, pp. 3943–3960.
[3] F. Fang, P. Stone, M. Tambe, When security games go green: designing defender strategies to prevent poaching and illegal fishing, in: IJCAI, AAAI Press, 2015, pp. 2589–2595.
[4] A. Sinha, F. Fang, B. An, C. Kiekintveld, M. Tambe, Stackelberg security games: looking beyond a decade of success, in: IJCAI, ijcai.org, 2018, pp. 5494–5501.
[5] J. Gan, E. Elkind, S. Kraus, M.J. Wooldridge, Defense coordination in security games: equilibrium analysis and mechanism design, Artif. Intell. 313 (2022) 103791.
[6] T.H. Nguyen, R. Yang, A. Azaria, S. Kraus, M. Tambe, Analyzing the effectiveness of adversary modeling in security games, in: AAAI, AAAI Press, 2013.
[7] J. Gan, H. Xu, Q. Guo, L. Tran-Thanh, Z. Rabinovich, M.J. Wooldridge, Imitative follower deception in Stackelberg games, in: EC, ACM, 2019, pp. 639–657.
[8] A. Alshamsi, F.L. Pinheiro, C.A. Hidalgo, Optimal diversification strategies in the networks of related products and of related research areas, Nat. Commun. 9 (1) (2018) 1328.
[9] J. Tsai, T.H. Nguyen, M. Tambe, Security games for controlling contagion, in: AAAI, AAAI Press, 2012.
[10] K.C. Nguyen, T. Alpcan, T. Basar, Security games with incomplete information, in: ICC, IEEE, 2009, pp. 1–6.

[11] C. Kiekintveld, M. Jain, J. Tsai, J. Pita, F. Ordóñez, M. Tambe, Computing optimal randomized resource allocations for massive security games, in: AAMAS (1), IFAAMAS, 2009, pp. 689–696.
[12] D. Korzhyk, V. Conitzer, R. Parr, Complexity of computing optimal Stackelberg strategies in security resource allocation games, in: AAAI, AAAI Press, 2010.
[13] M. Jain, D. Korzhyk, O. Vanek, V. Conitzer, M. Pechoucek, M. Tambe, A double oracle algorithm for zero-sum security games on graphs, in: AAMAS, 2011, pp. 327–334.
[14] M. Jain, V. Conitzer, M. Tambe, Security scheduling for real-world networks, in: AAMAS, IFAAMAS, 2013, pp. 215–222.
[15] Z. Wang, Y. Yin, B. An, Computing optimal monitoring strategy for detecting terrorist plots, in: AAAI, AAAI Press, 2016, pp. 637–643.
[16] J. Gan, B. An, Y. Vorobeychik, B. Gauch, Security games on a plane, in: AAAI, AAAI Press, 2017, pp. 530–536.
[17] Y. Vorobeychik, B. An, M. Tambe, S.P. Singh, Computing solutions in infinite-horizon discounted adversarial patrolling games, in: ICAPS, AAAI, 2014.
[18] Y. Yin, H. Xu, J. Gan, B. An, A.X. Jiang, Computing optimal mixed strategies for security games with dynamic payoffs, in: IJCAI, AAAI Press, 2015, pp. 681–688.
[19] R. Bai, H. Lin, X. Yang, X. Wu, M. Li, W. Jia, Defending against contagious attacks on a network with resource reallocation, Proc. AAAI Conf. Artif. Intell. 35 (2021).
[20] M. Li, L. Tran-Thanh, X. Wu, Defending with shared resources on a network, in: AAAI, AAAI Press, 2020, pp. 2111–2118.
[21] J. Aspnes, K.L. Chang, A. Yampolskiy, Inoculation strategies for victims of viruses and the sum-of-squares partition problem, in: SODA, SIAM, 2005, pp. 43–52.
[22] V.S.A. Kumar, R. Rajaraman, Z. Sun, R. Sundaram, Existence theorems and approximation algorithms for generalized network security games, in: ICDCS, IEEE Computer Society, 2010, pp. 348–357.
[23] D. Acemoglu, A. Malekian, A.E. Ozdaglar, Network security and contagion, J. Econ. Theory 166 (2016) 536–585.
[24] R. Bai, H. Lin, X. Yang, X. Wu, M. Li, W. Jia, Stackelberg security games with contagious attacks on a network: reallocation to the rescue, J. Artif. Intell. Res. 77 (2023) 487–515.
[25] B. An, et al., Security games with surveillance cost and optimal timing of attack execution, in: AAMAS, IFAAMAS, 2013, pp. 223–230.
[26] Y. Vorobeychik, J. Letchford, Securing interdependent assets, Auton. Agents Multi-Agent Syst. 29 (2) (2015) 305–333.
[27] J. Gan, B. An, Y. Vorobeychik, Security games with protection externalities, in: AAAI, AAAI Press, 2015, pp. 914–920.
[28] T. Kroupa, T. Votroubek, Multiple oracle algorithm to solve continuous games, in: GameSec, in: Lecture Notes in Computer Science, vol. 13727, Springer, 2022, pp. 149–167.
[29] N. Basilico, N. Gatti, F. Amigoni, Leader-follower strategies for robotic patrolling in environments with arbitrary topologies, in: AAMAS (1), IFAAMAS, 2009, pp. 57–64.
[30] J. Lou, Y. Vorobeychik, Equilibrium analysis of multi-defender security games, in: IJCAI, AAAI Press, 2015, pp. 596–602.
[31] J. Gan, E. Elkind, M.J. Wooldridge, Stackelberg security games with multiple uncoordinated defenders, in: AAMAS, International Foundation for Autonomous Agents and Multiagent Systems Richland, ACM, SC, USA, 2018, pp. 703–711.
[32] J. Gan, E. Elkind, S. Kraus, M.J. Wooldridge, Mechanism design for defense coordination in security games, in: AAMAS, International Foundation for Autonomous Agents and Multiagent Systems, 2020, pp. 402–410.
[33] D. Mutzari, Y. Aumann, S. Kraus, Robust solutions for multi-defender Stackelberg security games, in: IJCAI, ijcai.org, 2022, pp. 433–439.
[34] Z. Song, C.K. Ling, F. Fang, Multi-defender security games with schedules, in: International Conference on Decision and Game Theory for Security, Springer, 2023, pp. 65–85.
[35] C.-W. Shao, Y.-F. Li, C.-Y. Shen, S.-Z. Liu, Z. Yang, Risk-sharing mechanism design in non-cooperative multi-defender Stackelberg defense resources allocation game, IEEE Trans. Autom. Sci. Eng. (2023).
[36] V. Conitzer, T. Sandholm, Computing the optimal strategy to commit to, in: EC, ACM, 2006, pp. 82–90.
[37] I.P. Gent, T. Walsh, Phase transitions and annealed theories: number partitioning as a case study, in: ECAI, Citeseer, 1996, pp. 170–174.
[38] J. Leskovec, A. Krevl, SNAP datasets: Stanford large network dataset collection, http://snap.stanford.edu/data, Jun. 2014. (Accessed 1 January 2022).
[39] J. Zhang, Y. Wang, J. Zhuang, Modeling multi-target defender-attacker games with quantal response attack strategies, Reliab. Eng. Syst. Saf. 205 (2021) 107165.