AAAI Technical Report FS-12-08
Social Networks and Social Contagion

Security Games on Social Networks

Thanh H. Nguyen[1], **Jason Tsai**[1], **Albert Jiang**[1],
Emma Bowring[2], **Rajiv Maheswaran**[1], **Milind Tambe**[1]

[1]University of Southern California, Los Angeles, CA 90089
{thanhhng, jasontts, jiangx, maheswar, tambe} @usc.edu

[2]University of the Pacific, Stockton, CA 95211
ebowring@pacific.edu

Abstract

Many real-world problems exhibit competitive situations in which a defender (a defending agent, agency, or organization) has to address misinformation spread by its adversary, e.g., health organizations cope with vaccination-related misinformation provided by anti-vaccination groups. The rise of social networks has allowed misinformation to be easily and quickly diffused to a large community. Taking into account knowledge of its adversary's actions, the defender has to seek efficient strategies to limit the influence of the spread of misinformation by the opponent.

In this paper, we address this problem as a blocking influence maximization problem using a game-theoretic approach. Two players strategically select a number of seed nodes in the social network that could initiate their own influence propagation. While the adversary attempts to maximize its negative influence, the defender tries to minimize this influence. We represent the problem as a zero-sum game and apply the Double Oracle algorithm to solve the game in combination with various heuristics for oracle phases. Our experimental results reveal that by using the game theoretic approach, we are able to significantly reduce the negative influence in comparison to when the defender does not do anything. In addition, we propose using an approximation of the payoff matrix, making the algorithms scalable to large real-world networks.

Introduction

With an increasing number of users, online social networks have become important and efficient media for disseminating information to a large number of people. People tend to be influenced by their social friends in deciding whether to adopt an innovation such as a political idea or a new product. Many diverse topics in the research areas of sociology, economics, epidemiology, and computer science closely relate to the social network domain such as effects of advertising, epidemics and spread of certain social behavioral patterns (Achrekar et al. 2011), (Fu et al. 2011), (Trusov, Bucklin, and Pauwels 2009).

Although social networks are considered as the main news resource for many people today, there is no guarantee of correctness for the information. Misinformation available in social networks could cause a great negative effect in the large population. There are examples of this phenomenon in the health domain in which vaccination misinformation is prevalent in social networks, partially accounting for the decreasing rate of vaccination (Shetty 2010). In fact, in many cases, misinformation is intentionally spread by groups which oppose vaccination (Davies, Chapman, and Leask 2002), (Kata 2011). In the presence of misinformation, it is important to find out an efficient way to diffuse 'good' information to fight against the misinformation campaigns.

Many researchers have addressed this scenario as a competitive influence problem, seeking strategies to efficiently limit misinformation propagated by opponents (Borodin, Filmus, and Oren 2010), (Budak, Agrawal, and El Abbadi 2011). In their work, they make a strong assumption that opponents are static; meaning that their actions are already known and fixed. In real-world problems, however, the presence of intelligent adversaries who act strategically against actions of the defender makes the problem much harder to solve. This scenario represents the situation that anti-vaccination activists actively spread misinformation in social networks in the attempt of preventing people from getting vaccines (Smith 2010) (Kata 2011).

Motivated by this scenario, the key novelty of this paper is the application of game theoretic approach to analyse the problem in combination with influence diffusion models in social networks. We model the problem as a zero-sum game in which each player chooses a set of 'source' nodes to propagate their own influence simultaneously. While the adversary attempts to maximize their negative influence, the defender spreads their positive influence to block that negative influence. We adopt the Linear Threshold Model (LTM) which emphasizes strong relations between people influencing their common social friends (Kempe, Kleinberg, and Tardos 2003) to examine the influence cascades in social networks.

This game-theoretic model is computationally challenging because of the large number of strategies and the difficulty in computing payoffs. To cope with the problem of the large number of strategies, similar to (Tsai, Nguyen, and Tambe 2012), we apply the Double Oracle algorithm to solve the problem. In short, the Double Oracle algorithm computes an exact Nash equilibrium of the game using only a small portion of the payoff matrix (each element of the matrix is the payoff for a pair of players' actions). However, in their work, they haven't dealt with the problem of the difficulty in computing payoffs, which makes their algorithms infeasible for large games. In our works, we propose using the LDAG approach (Chen, Yuan, and Zhang 2010) for approximating the payoff matrix which outperforms their Monte-Carlo approach in runtime. Finally, we applied several ways to approximate the oracle phases: 1) the GreedyMC heuristic which guarantees a good solution (Kempe, Kleinberg, and Tardos 2003) but runs slowly, 2) the GreedyLDAG heuristic which shows good results in the case of fixed attacker actions (He et al. 2011), and 3) our new-PageRank originated from the PageRank algorithm (Page et al. 1999) which provides a comparable solution to the GreedyMC while runs faster. Extensive experiments were conducted with both synthetic and real-world graph data in order to analyse both runtime efficiency and solution quality of these algorithms.

Domain Example - Vaccination Misinformation Problem

This section outlines a motivating domain as an example, however, our techniques may generalize to several domains; indeed, the rest of the paper will use a more general domain.

Vaccines are one of the most important and efficient methods for people to protect themselves from diseases. However, a vast amount of negative vaccination information available in social networks is negatively affecting people's perception of vaccinations. Approximately 25.3% of the HPV-related video clips on Youtube potrayed the HPV vaccination negatively (Ache and Wallace 2008) and 43% of MySpace blogs which contain HPV information are classified as negative (Keelan et al. 2010).

There exist anti-vaccination groups who are intentionally spreading vaccination misinformation. Anti-vaccination lobbying organizations such as the US National Vaccine Information Centre (NVIC) and the Coalition for SafeMinds have become so highly organised that they threaten vaccination rates (Shetty 2010). Health agencies such as the European Society of Clinical Microbiology and Infectious Diseases (ESCMID) and World Health Organization (WHO) have become increasingly concerned about the vaccination misinformation that anti-vaccination groups are spreading (Shetty 2010).

It's important for health experts to figure out an efficient way to communicate with the public to fight against anti-vaccination messages. Both the messenger and the way the message is communicated are essential for the message to be effective. In fact, it is impossible for supporters to persuade all users of social media directly due to the budget and time constraint. Instead, strategists attempt to point out a small number of early 'positive' adopters in order to trigger a large cascade of further 'good' adoptions, to minimize the number of users who adopt negative information.

The vaccination problem can be modelled as a misinformation blocking problem in which vaccination supporters attempt to spread correct vaccination information to limit the influence of incorrect information provided by vaccination opponents. Vaccination supporters are presented as the defender that attempts to find a set of *source* nodes to start spreading positive influence (correct information) from in order to minimize the number of negatively influenced nodes (users who adopt negative information). Anti-vaccination groups are defined as the adversary who are finding a set of nodes to spread negative influence (incorrect information) to maximize the number of such nodes.

Related Work

The problem of finding a small set of influential nodes in a social network such that their aggregated influence in the network is maximized is called the Influence Maximization Problem (*IMP*). In *IMP*, a social network is represented as a directed graph in which nodes are users and edges are connections between users; orientations of edges imply the direction of influence. Each edge is associated with a weight representing influence strength between the users. Finding exact optimal solutions for *IMPs* is computationally difficult (it is NP-hard for most models that have been studied (Kempe, Kleinberg, and Tardos 2003)) and has motivated researchers to look for various approximations that run faster while still keeping a high solution quality.

Kempe proposed a greedy algorithm which guarantees an $(1 - \frac{1}{e})$-approximation for *IMPs* (Kempe, Kleinberg, and Tardos 2003). Following this study, researchers have investigated different techniques for solving *IMPs* where they utilize the local structures of networks to estimate the influence of nodes over the networks (Chen, Wang, and Wang 2010), (Chen, Yuan, and Zhang 2010). In addition to seeking new techniques, other studies looked at variances of *IMPs* such as the Competitive Maximization Problem that addresses the case of two players competing with each other in order to maximize their own influence (Bharathi, Kempe, and Salek 2007), or the case where one player tries to limit his / her competitor's influence (He et al. 2011). Nevertheless, none of these studies address the context where both players are strategic, and always assume a fixed action for one player. In our study, we address the influence blocking problem where both the players are strategic: we model the *IMP* as a zero sum game and assume that the adversary responds optimally against the defender's action.

For large-graph zero-sum games, (Jain et al. 2011) used a Double Oracle approach to iteratively generate the set of actions for both the defender and attacker to determine an exact Nash equilibrium of the game. This approach overcomes the computational barriers of traditional algorithms for zero-sum security games which require generating the whole action space for both players.

(Tsai, Nguyen, and Tambe 2012) applied the Double Oracle approach to solve contagion games in the counterinsurgency domain. In their study, the Independent Cascade Model (ICM) (Kempe, Kleinberg, and Tardos 2003) is used to present the influence diffusion process in networks. In short, ICM emphasizes the role of the active nodes that spread influence to other nodes; each active node can influence other inactive nodes independently from other active nodes. In this paper, we look at the Linear Threshold Model (LTM) (Kempe, Kleinberg, and Tardos 2003) which focuses on the role of inactive nodes: each inactive node is activated only when there is a sufficient number of its neighbors already activated. These two models could be applied to different domains. For example, ICM was investigated in the context of marketing (Goldenberg, Libai, and Muller 2001) and LTM is the general model of the models studied in the sociology (Granovetter 1978). Moreover, they use the exact payoff matrix, which works well with their domain's networks. This is also inapplicable for our domain because of runtime issues in large social networks. Our approach of using an approximation of the payoff matrix overcomes this runtime barrier.

Problem Definition

We represent a social network as a directed graph $G(V, E)$ where V is the set of nodes and E is the set of edges in the graph. Each node u in the graph represents a user and each edge (u, v) represents an influence between two users in the network. The direction of each edge infers the orientation of influence. Each edge $e_{u,v} = (u, v)$ in G is associated with a weight $w_{u,v}$ measuring the influence of user u on user v.

We model the misinformation blocking problem as a zero-sum game in which a defender's purpose is to minimize the number of negatively activated nodes and an attacker aims to maximize the number of such nodes. Each node in the graph has three different states: negative, neutral, and positive. Initially, the defender can choose a subset of their 'source' nodes (positively activated nodes), $S_d \subseteq V$, and the attacker can select a subset of their 'source' nodes (negatively activated nodes), $S_a \subseteq V$, to start spreading their influence simultaneously. Nodes other than the source nodes stay neutral in the beginning of the influence diffusion process. The payoff for the attacker, u_a, is the expected number of nodes influenced by that player while the defender's payoff, u_d, is the opposite of the attacker's payoff, $-u_a$. The objective is to find an optimal mixed strategy for the defender given that the attacker will respond optimally.

We use the commonly used and fundamental Linear Threshold Model which is described in the next section to represent the influence spreading process in our problem.

Influence Diffusion Model

In LTM, a node u activates its neighbor v in correlation with v's other neighbors. Initially, each node v is associated with an activation threshold θ_v which is chosen uniformly at random in $[0, 1]$ representing the weighted fraction of v's neighbors that must be positively (negatively) activated in order for v to be positively (negatively) activated. The diffusion process is unfolded in discrete time steps. Given a set of 'source' nodes, an inactive node v can be activated only when the weighted sum of its activated neighbors exceeds the threshold θ_v. The total sum of weights of edges ending at v, $\sum_u w_{uv}$, satisfies $\sum_u w_{uv} \leq 1$. The activation process stops when no more nodes can be activated.

In our competitive diffusion model, two players disseminate their influence simultaneously from their 'source' nodes. When a node is activated by both players at the same time, the defender has probability p of defeating the attacker and thus activating that node. More specifically, in LTM, if the weighted sum of positively activated neighbors and weighted sum of negatively activated neighbors of an inactive node w exceeds the positive threshold θ_w^+ and negative threshold θ_w^- respectively at the same time, then w has probability p of being positively and probability $(1 - p)$ of being negatively activated. In this scenario, each directed edge (u, v) is associated with two weights: $w^+(u, v)$ and $w^-(u, v)$ that correspond to the positive and negative influence of the node u on node v respectively. This assumption makes sense as in general, $w^+(u, v)$ is not always necessarily equal to $w^-(u, v)$.

Computing the Payoff Matrix

Exact Payoff: Under the Linear Threshold Model in which the influence propagation process is described as a probabilistic process, for each pair of actions of players, their payoffs can be computed using the Monte Carlo simulation (Kempe, Kleinberg, and Tardos 2003). However, the Monte Carlo simulation requires generating thousands of samples which is time-consuming to exactly compute payoff values. Therefore, we propose using the LDAG (Local Directed Acyclic Graph) influence model to compute the payoff matrix in which influence to each node is estimated locally in its neighborhood area (Chen, Yuan, and Zhang 2010).

Approximated Payoff: In a single LDAG model in which there is one single player, it computes a local DAG (Directed Acyclic Graph) for every node u in the graph which covers a significant portion of influence from other nodes to u such that each node $v \in LDAG(u)$ must satisfy $Inf(v, u) \geq \theta$ where θ is a given threshold and $Inf(v, u)$ is a measure of the influence from v to u. The activation probability of a node u, denoted by $ap(u)$, can be computed by simply aggregating the activation probabilities of its neighbors in its LDAG: $ap(u) = \sum_{v \in N_{in}(u)} w_{v,u} ap(v)$. This property allows a linear computation of the activation probability of a node within its local DAG (see (Chen, Yuan, and Zhang 2010) for more details of computing $Inf(v, u)$ and $LDAG(u), \forall u, v \in V$). This activation probability value could be used to approximate the payoffs.

In the competitive influence problem, however, both the negative and positive influence are diffused simultaneously and interfere with each other. A new competitive model which utilizes LDAG is proposed by (He et al. 2011) to deal with this problem. In their study, they assume that when a node is activated by both players at the same time, the attacker will have the advantage over the defender. Yet, it is not difficult to adjust their model for our general case. For each pair of actions, the adversary's payoff would be $\sum_{u \in V} ap^-(u)$ and the defender's payoff is $-\sum_{u \in V} ap^-(u)$. By using the negative activation probabilities of nodes, we can approximate the payoff matrix that overcomes the runtime issue of the exact computation of the payoffs. For more details about computing $ap^{-}(u)$, see (He et al. 2011).

Double Oracle Approach

The most common approach to solve a zero-sum game is the naive *Maximin* algorithm. In general, *Maximin* computes a Nash equilibrium for the game. Although it gives an exact solution for the game, it suffers from both memory and runtime barriers because of its requirement of enumerating the whole payoff matrix. Specifically, the size of the payoff matrix increases exponentially with the size of the graph. For example, if the number of nodes in the graph, N, was equal to 1000, and the number of defender resources $|S_d|$ was equal to 30, there will be $\binom{1000}{30}$ actions of the defender. Therefore, computing the payoff matrix for a large game is a computational limitation.

The Double Oracle algorithm, in contrast, deals with only a small portion of the players' action space while still guaranteeing to generate an exact Nash equilibrium for the game (McMahan, Gordon, and Blum 2003), (Jain et al. 2011). Thus, it is very efficient to solve large real-world games. The core part of the Double Oracle algorithm is the *Maximin* algorithm. Instead of generating the whole action space for both players, it iteratively adds a new action into the current set of chosen actions in the defender and adversary oracle phases. The Double Oracle algorithm is described in Algorithm 1. In this algorithm, D and A respectively denote the sets of actions of the defender and the attacker generated so far. The *Maximin* computes equilibrium mixed strategies for the defender and adversary given the payoff matrix $P(D, A)$ computed either exactly or approximately over D and A. The outcome of *Maximin* is a probability distribution (x_D, x_A) over D and A, respectively. In the defender oracle phase, a new defender action is generated which is an optimal pure strategy for the defender against the attacker's mixed strategy (A, x_A) given by *Maximin*. Similarly, the adversary oracle phase generates a new optimal action for the attacker against (D, x_D). The process stops when no new actions are generated.

Algorithm 1 DOUBLE ORACLE ALGORITHM

```
1: Initialize D with random defender allocations.
2: Initialize A with random attacker allocations.
3: repeat
4:     $(x_D, x_A)$ = Maximin(D,A, P(D, A))
5:     $\mathbf{D} = \mathbf{D} \cup \{\text{DefenderOracle}(A, x_A)\}$
6:     $\mathbf{A} = \mathbf{A} \cup \{\text{AttackerOracle}(D, x_D)\}$
7: until convergence
8: return $(D, x_D), (A, x_A)$
```

Maximin

In the Double Oracle algorithm, the *Maximin* algorithm solves for the optimal strategies for both players given the set of strategies generated from oracle phases so far. Specifically, finding an optimal mixed strategy of the defender can be described as the following Maximin linear program::

$$\text{maximize}_{x_D} r_D$$
$$\text{subject to } r_D \leq \sum_d x_d \cdot r_d(d, a), \forall a \in A$$
$$\sum_{d \in D} x_d = 1$$

where $r_d(d, a)$ is the defender's payoff given a defender's pure strategy d and an adversary's pure strategy a. The linear program for finding an optimal mixed strategy for the adversary is similar.

Defender Oracle Phase

The defender oracle phase computes the optimal pure strategy of the defender against the mixed strategy of the attacker generated by *Maximin*. In the following, we address two cases: 1) the Exact Oracle which computes the defender optimal response, yet copes with low runtime efficiency, and 2) the Approximated Oracle that generates a good approximation for the defender's optimal action and runs much faster.

Exact Oracle: In the exact defender oracle phase, we look for an optimal pure strategy of the defender in the whole strategy space of the defender. Using the exact oracle phase provides the best action for the defender. Nevertheless, the defender's action space is very large even in small graphs. For example, given a graph with $N = 100$ and $|S_d| = 3$, the number of pure strategies for the defender is $\binom{100}{3} \approx 1.62 \cdot 10^5$. This exponentially large pure strategy space significantly deteriorates the algorithm's runtime performance.

To overcome this issue, we propose using an approximation for the defender oracle phase. Details are described in the following:

Approximated Oracle In the approximated oracle, we propose using three heuristics to estimate the defender pure action: **GreedyMC**, **NewPageRank** and **GreedyLDAG**.

GreedyMC: We use the greedy algorithm proposed by Kempe (Kempe, Kleinberg, and Tardos 2003) to estimate the best action for the defender against the attacker's mixed strategy generated by *Maximin*. In short, the main idea of GreedyMC is to iteratively add a new node into the defender's list of seed nodes until the size of the seed set reaches the number of defender resources. The algorithm is described in Algorithm 2.

Let $\gamma(d, a)$ be the set of nodes u, which are negatively activated if a is a pure strategy of the attacker and no strategy of the defender and are not negatively activated if a is a pure strategy of the attacker and d is a pure strategy of the defender. This $\gamma(d, a)$ is the set of nodes that have been blocked from the negative influence. Let $\sigma(d, a)$ be the cardinality of $\gamma(d, a)$ and $\sigma(d, (A, x_A)) = \sum_a x_a \cdot \sigma(d, a)$ be the expected number of blocked nodes over the adversary's mixed strategy (A, x_A).

Under LTM, it is demonstrated that if $\sigma(d, (A, x_A))$ is monotone and submodular,[1] the GreedyMC algorithm guarantees that $\sigma(d, (A, x_A)) \geq (1 - \frac{1}{e})\sigma(d^*, (A, x_A))$ where

[1]**Sub-modularity** property: Given a set U, a function $f : U \rightarrow$

Algorithm 2 GreedyMC Algorithm, (A, x_A)

```
1: Initialize defender action $d = \emptyset$
2: repeat
3:     $v = max_u(\sigma(d, (A, x_A)) - \sigma(d \cup \{u\}, (A, x_A)))$
4:     $d = d \cup \{v\}$
5: until $|d| > numRes$, where $numRes$ is number of defender 'source' nodes.
6: return $d$
```

d^* is the optimal pure strategy of the defender. Then applying GreedyMC to the defender oracle phase can achieve an approximation ratio of $(1 - \frac{1}{e})$ for the optimal expected number of blocked nodes (Tsai, Nguyen, and Tambe 2012), i.e., $\sigma((D, x_D), (A, x_A)) \geq (1 - \frac{1}{e})\sigma((D^*, x_{D^*}), (A^*, x_{A^*}))$ where $(D, x_D), (A, x_A)$ are the players' mixed strategies returned by the Double Oracle algorithm using GreedyMC for the defender phase and (D^*, x_{D^*}), (A^*, x_{A^*}) is the exact Nash equilibrium solved by the algorithm using the exact defender phase.

(He et al. 2011) provides a thorough proof for the submodularity property of function $\sigma(d, (A, x_A))$ in the case that negative influence always wins positive influence when both influences activate a node at the same time. Following their proof, this property can be easily demonstrated in our general case.

NewPageRank: Although GreedyMC provides a good solution quality, however, it still uses the Monte-Carlo simulation for its computation which is time-consuming. PageRank is the algorithm used by Google search engine (Page et al. 1999). The fundamental idea of PageRank is that the importance of a web page can be determined by the links pointing to it from other web pages, which means that the importance of a page is increased by the number and rank of sites that link to it. Thus, the rank of a page P, denoted $r(P)$, is defined as the following: $r(P) = \sum_Q \frac{r(Q)}{|Q|}$ where a page Q pointing to the page P and $|Q|$ is the number of pages pointing to Q.

In LTM, the importance of every node in a graph is 'voted' by its out-neighbors (i.e., set of nodes which can be influenced by that node) which is very similar to to the core of PageRank. Motivated by this similarity, we propose the newPageRank algorithm to apply into the oracle phases: For each node u, its negative rank $r^-(u)$ is computed as follows: $r^-(u) = \sum_{v \in N_{out}(u)} s^-(u, v)r^-(v)$ where $s^-(u, v)$ is the normalized weight of the edge (u, v), $s^-(u, v) = \frac{w^-(u,v)}{\sum_{u^* \in N_{in}(v)} w^-(u^*,v)}$. The positive rank of each node is computed similarly. These ranks infer how important each node is in influencing other nodes in the graph. The newPageRank algorithm is given in Algorithm 3.

Given that u is a 'source' negative node, Line (6) of Algorithm 3 means that all of its in-neighbors can not pass their negative influence through that node, Line (9) means that if a node $v \in N_{out}(u)$ is positively activated, the defender has successfully blocked negative influence from u passing to v.

R is sub-modular iff $f(S \cup \{v\}) - f(S) \geq f(T \cup \{v\}) - f(T)$ where $S \subseteq T$.

Algorithm 3 newPageRank Algorithm, (A, x_A), $r^-(u), u \in V$

```
1: Initialize defender action $d = \emptyset$,
2: Initialize blocking value $b(u) = 0, \forall u \in V$
3: for all $a \in A$ do
4:     for all node $u \in a$ do
5:         Initialize new negative rank $r^*(v) = r^-(v), \forall v \in V$
6:         Compute the new ranks $r^*(v) = r^*(v) - w^-(v, u)r^-(u), \forall v \in N_{in}(u)$
7:     end for
8:     for all node $u \in a$ do
9:         $b(v) = b(v) + x_a w^-(u, v)r^*(v), \forall v \in N_{out}(u), v \notin a$
10:        $b(u) = b(u) + x_a r^*(u)p$
11:    end for
12: end for
13: $d = topK_u b(u), u \in V$
14: return $d$
```

Similarly, Line (10) means that if u is activated by both the defender and adversary at the same time, the defender has a chance of p to block the negative influence that u can spread. The blocking value of each node v, $b(v)$, is the measurement of the negative influence that is blocked by v if v is chosen as a positive 'source' node. Finally, the set of nodes that have the highest blocking values is chosen to be the pure strategy of the defender in Line (13).

GreedyLDAG: Motivated by the LDAG model, a new algorithm to approximate the defender oracle phase is proposed by He et al. (2011) which outperforms GreedyMC in runtime while still guaranteeing a good solution quality in practice. Although their algorithm is applied for dealing with an adversary's fixed pure strategy, we expand it to the case of a mixed strategy of the adversary (A, x_A).

Let $OutS^+(u)$ be the set of nodes that node u can influence and $\delta(u, x, (d, a)) = ap^-(x, (d, a)) - ap^-(x, (d \cup \{u\}, a))$ be the amount of negative influence to the node x that is blocked by adding the node u into the defender action d. Let $DecInf(u)$ be the total amount of negative influence blocked by adding u: $DecInf(u) = \sum_{a \in A} x_a \sum_{x \in OutS^+(u)} \delta(u, x, (d, a)) = \sum_{x \in OutS^+(u)} E_{a \in A}\delta(u, x, (d, a))$

GreedyLDAG, given in Algorithm 4, can be summarized as follows: iteratively add a new node u with the highest $DecInf(u)$ value to the set of 'seed' nodes of the defender. Yet instead of recomputing $DecInf(u)$ for every $u \notin d$ in every iteration, GreedyLDAG makes updates only for the nodes u that are affected by the newly added node to d in previous iteration (Lines (7-8)).

Attacker Oracle Phase

Similar to the defender oracle phase, we can use both the exact and approximated computations for the attacker's best response against the mixed strategy of the defender generated using *Maximin*. However, the attacker's objective function is different as the attacker attempts to maximize the number of negatively activated nodes.

The newPageRank algorithm in the adversary oracle phase is similar to the defender oracle phase, except for

Algorithm 4 GREEDYLDAG ALGORITHM, (A, x_A)

```
1: Initialize defender action $d = \emptyset$
2: Compute $DecInf(u)$ for every $u$
3: repeat
4:     $v = argmax_u DecInf(u)$
5:     for $x \in OutS^{+}(v)$ do
6:         for $u \in LDAG^{+}(x)$ do
7:             $DecInf(u) = DecInf(u) - E_{a \in A}[\delta(u, x, (d, a))]$
8:             $DecInf(u) = DecInf(u) + E_{a \in A}[\delta(u, x, (d \cup \{v\}, a))]$
9:         end for
10:    end for
11:    $d = d \cup \{v\}$
12: until $|d| > numRes$, where $numRes$ is number of defender's source nodes.
13: return $d$
```

Lines (8)-(11), which are replaced by $\forall u \in V, v(u) = v(u) + x_d r^*(u), u \notin d$ and $v(u) = v(u) + x_d(1 - p)r^*(u), u \in d$ respectively. It measures how much negative influence the node u can diffuse if it is chosen as a negative 'source' node. Finally, we can apply approximations for the both oracle phases, which speeds up the double oracle algorithm.

Experiments

We evaluate both the runtime efficiency and solution quality for the algorithms in this section.

Data Sets

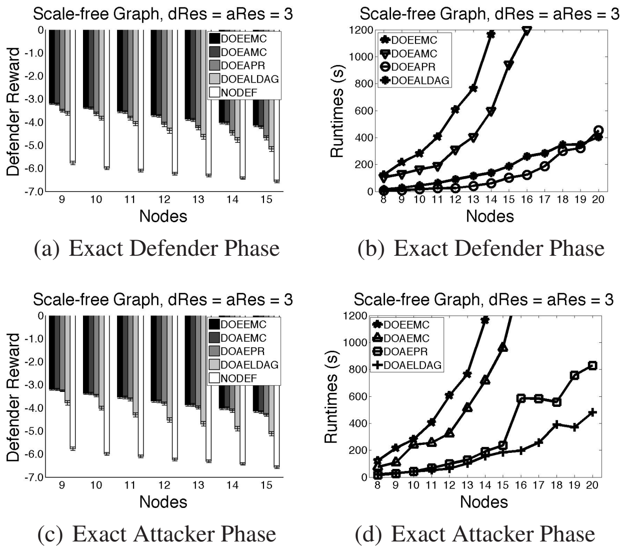
For these experiments, we use two data sets on graphs: (1) synthetically generated scale-free graphs, and (2) real network obtained from the caHepTh graph (Les 2012). A scale-free network is a connected graph with the property that the distribution of node degrees follows a power law. Scale-free graphs have been commonly used as proxies for real-world networks because node degrees in many real-world graphs exhibit a power law distribution (Clauset, Shalizi, and Newman 2007). On the other hand, the caHepTh graph represents the collaboration network of Arxiv High Energy Physics Theory containing 9877 nodes and 51971 edges.

In our settings, random graphs of various sizes were generated which are either scale-free graphs or extracted subgraphs of the caHepTh network. We ran 30 trials for each graph size. Each edge of the graph, $e_{u,v} = (u, v)$, is assigned a weight $w^{+}_{u,v} = w^{-}_{u,v} = \frac{1}{indeg(v)}$, where $indeg(v)$ is the in-degree of the node v. We set $p = 0.5$, meaning that negative and positive influence have 50/50% chance to win the opposite side in every simultaneous activation. We ran 30000 Monte Carlo simulations per estimation.

We applied different heuristics for both the defender and attacker oracle phases. The combination of oracle phases using different heuristics generated 12 variants of the Double Oracle algorithm. The list of these 12 variants are provided in the Table 1.

Algo Label	Def. Oracle	Att. Oracle
DOEEMC	EXACT	EXACT
DOAEMC	GREEDYMC	EXACT
DOEAMC	EXACT	GREEDYMC
DOAAMC	GREEDYMC	GREEDYMC
DOAEPR	NEWPAGERANK	EXACT
DOEAPR	EXACT	NEWPAGERANK
DOAAPR	NEWPAGERANK	NEWPAGERANK
DOAELDAG	GREEDYLDAG	EXACT
DOEALDAG	EXACT	GREEDYLDAG
DOAALDAG	GREEDYLDAG	GREEDYLDAG
MCPR	GREEDYMC	NEWPAGERANK
PRMC	NEWPAGERANK	GREEDYMC

Table 1: Algorithms evaluated



(a) Exact Defender Phase (b) Exact Defender Phase

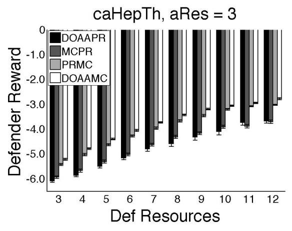
(c) Exact Attacker Phase (d) Exact Attacker Phase

Figure 1: Performance on Scale-free graphs

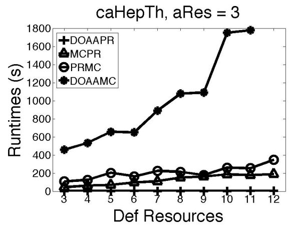
Results

In the following, we first conducted numerous experiments using the MCPayoff matrix (exact payoff matrix) to examine the performance of several heuristics applied for oracle phases and finally we compare the performance of the algorithm between using MCPayoff matrix and using LDAG-Payoff matrix (approximated payoff matrix).

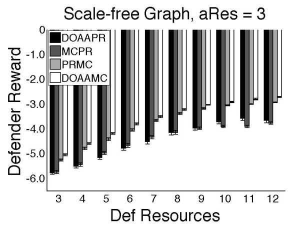
MCPayoff: In the Figure 1, the x-axis is the size of graphs and the y-axis is the average reward of the defender (for the reward graphs) or the average runtime in seconds (for the runtime graphs). The defender's reward is computed based on the mixed strategy of the defender generated by the Double Oracle algorithm and the optimal pure strategy of the adversary responding against the defender's strategy. We conducted experiments with one approximated oracle phase on small scale-free graphs of 8 to 20 nodes. It shows that using the double oracle approach could considerably limit the negative influence of the attacker when compared to a defender who does not execute any action (labeled NODEF in Figure 1). In addition, although the GreedyMC provides a better solution as compared to the newPageRank



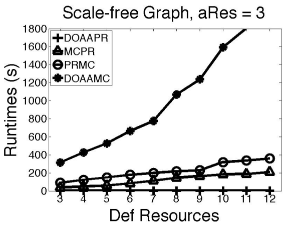
(a) caHepth, Def Reward



(b) caHepth, Runtime

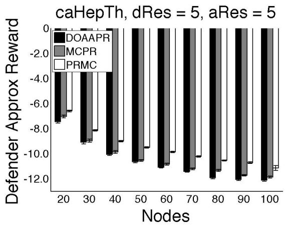


(c) ScaleFree, Def Reward

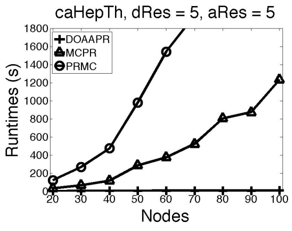


(d) ScaleFree, Runtime

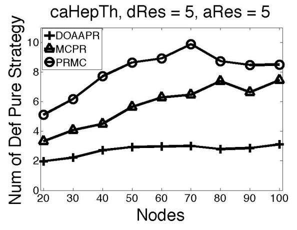
Figure 2: Performance on subgraphs of 25 nodes



(a) Defender Reward

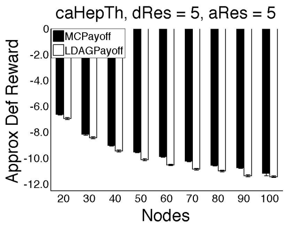


(b) Runtime

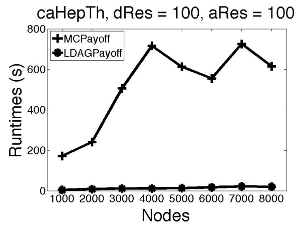


(c) Number of Defender Pure Strategies

Figure 3: Heuristic performance on the caHepTh subgraphs



(a) Defender Reward, PRMC algorithm



(b) Runtime, DOAAPR algorithm

Figure 4: Performance comparison between the MCPayoff and the LDAGPayoff matrix on the caHepTh subgraphs

and GreedyLDAG heuristics, its runtime grows very quickly even in small graph sizes (It reaches 20 minutes when the size of graphs is 16 nodes). Overall, the newPageRank outperforms other heuristics in the consideration of the tradeoff between the runtime and the solution quality.

In order to efficiently block the negative influence, the defender could use a large number of resources, however, increasing the number of defender resources will deteriorate the runtime of the algorithm. To analyze the trade-off between the defender reward and the runtime of the algorithms, we generated experiments with subgraphs of 25 nodes extracted from both random scale-free and caHepTh graphs. These results are shown in Figure 2, where the x-axis is the number of defender resources and the y-axis is the runtime in seconds or the defender reward. In these experiments, four algorithms were used: DOAAPR, MCPR, PRMC, DOAAMC. These four algorithms are chosen as they run fast as well as generate a high solution quality. It shows that DOAAMC is the most sensitive in runtime toward the number of defender resources. It means that GreedyMC runtime is considerably affected by the number of defender resources. In addition, using more defender resources definitely blocks the negative influence more efficiently (the defender reward increases steadily in the graphs).

From previous experiments, it is shown that the newPageRank heuristic could give us a high solution quality and low runtime consumption. However, it can be seen that applying newPageRank into the defender and attacker oracle phases could provide very different results. In the following experiment, we investigate the difference in performance between the defender and the attacker oracle phases under the same heuristics.

In Figure 3, experiments were conducted in subgraphs with size 20 to 100 nodes extracted from the caHepTh graph. The number of defender and attacker resources are dRes = aRes = 5. We examined three algorithms: DOAAPR, MCPR, and PRMC. These algorithms are chosen because they can run on large data sets. In this experiment, we approximated the reward of the defender, i.e., using the GreedyMC heuristic to compute the pure strategy of the attacker against the mixed strategy of the defender generated by the algorithms. It can be clearly seen in Figure 3a and Figure 3b that the defender reward in the MCPR case is worse than in the PRMC, however, MCPR runs much faster as compared to PRMC. It can be inferred that newPageRank gives a better estimation of the influence blocking values (in the defender oracle phase) than the influence spreading values (in the attacker oracle phase) of every node in graphs. In addition, Figure 3c shows that the number of defender pure strategies generated by the Double Oracle algorithm using PRMC is more than using MCPR. In other words, the oracle algorithm converges more quickly in MCPR than in PRMC. Finally, applying the newPageRank heuristic on both the oracle phases could provide a very fast algorithm.

LDAGPayoff: Although applying the newPageRank heuristic on both oracle phases could considerably reduce the runtime consumption, using the MCPayoff in the *Max-*

imin part still makes the algorithms run slowly. In our final experiment, we evaluate the performance of using LDAGPayoff matrix in large graphs. Figure 4a shows that the PRMC algorithm generates a slightly worse defender reward for LDAGPayoff as compared to when using MCPayoff. The defender reward is computed in the same way as in the previous experiment. Yet, the required runtime is reduced extremely in the LDAGPayoff case which is shown in the Figure 4b. Furthermore, the DOAAPR algorithm takes only 20 seconds in LDAGPayoff while 10 minutes in MCPayoff in average to finish computation in a 8000 node graph.

Conclusion

This study has applied a game-theoretic approach with the influence blocking mechanism of the LTM model on social networks. We present an algorithm based on double oracle approaches that can compute solutions to problems of real-world sizes. We presented 3 heuristics: GreedyMC, GreedyLDAG, and newPageRank and explored 12 combinations of these heuristics for solving this problem. Our experimental results show that the newPageRank heuristic outperforms the others in balancing between solution quality and runtime efficiency. Finally, the approach of using the payoff approximation opens a new efficient way to solve contagion games in large social networks.

Acknowledgement

This research is supported by MURI grant W911NF-11-1-0332.

References

[Ache and Wallace 2008] Ache, K., and Wallace, L. 2008. Human papillomavirus vaccination coverage on youtube. *American journal of preventive medicine* 35(4):389–392.

[Achrekar et al. 2011] Achrekar, H.; Gandhe, A.; Lazarus, R.; Yu, S.; and Liu, B. 2011. Predicting flu trends using twitter data. In *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on*, 702–707. IEEE.

[Bharathi, Kempe, and Salek 2007] Bharathi, S.; Kempe, D.; and Salek, M. 2007. Competitive influence maximization in social networks. *Internet and Network Economics* 306–311.

[Borodin, Filmus, and Oren 2010] Borodin, A.; Filmus, Y.; and Oren, J. 2010. Threshold models for competitive influence in social networks. *Internet and Network Economics* 539–550.

[Budak, Agrawal, and El Abbadi 2011] Budak, C.; Agrawal, D.; and El Abbadi, A. 2011. Limiting the spread of misinformation in social networks. In *Proceedings of the 20th international conference on World wide web*, 665–674. ACM.

[Chen, Wang, and Wang 2010] Chen, W.; Wang, C.; and Wang, Y. 2010. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 1029–1038. ACM.

[Chen, Yuan, and Zhang 2010] Chen, W.; Yuan, Y.; and Zhang, L. 2010. Scalable influence maximization in social networks under the linear threshold model. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, 88–97. IEEE.

[Clauset, Shalizi, and Newman 2007] Clauset, A.; Shalizi, C.; and Newman, M. 2007. Power-law distributions in empirical data. *Arxiv preprint arxiv:0706.1062*.

[Davies, Chapman, and Leask 2002] Davies, P.; Chapman, S.; and Leask, J. 2002. Antivaccination activists on the world wide web. *Archives of disease in childhood* 87(1):22–25.

[Fu et al. 2011] Fu, F.; Rosenbloom, D.; Wang, L.; and Nowak, M. 2011. Imitation dynamics of vaccination behaviour on social networks. *Proceedings of the Royal Society B: Biological Sciences* 278(1702):42–49.

[Goldenberg, Libai, and Muller 2001] Goldenberg, J.; Libai, B.; and Muller, E. 2001. Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review* 9(3):1–18.

[Granovetter 1978] Granovetter, M. 1978. Threshold models of collective behavior. *American journal of sociology* 1420–1443.

[He et al. 2011] He, X.; Song, G.; Chen, W.; and Jiang, Q. 2011. Influence blocking maximization in social networks under the competitive linear threshold model technical report. *Arxiv preprint arXiv:1110.4723*.

[Jain et al. 2011] Jain, M.; Korzhyk, D.; Vanek, O.; Conitzer, V.; Pechoucek, M.; and Tambe, M. 2011. A double oracle algorithm for zero-sum security games on graphs. In *AAMAS*, 327–334.

[Kata 2011] Kata, A. 2011. Anti-vaccine activists, web 2.0, and the postmodern paradigm–an overview of tactics and tropes used online by the anti-vaccination movement. *Vaccine*.

[Keelan et al. 2010] Keelan, J.; Pavri, V.; Balakrishnan, R.; and Wilson, K. 2010. An analysis of the human papilloma virus vaccine debate on myspace blogs. *Vaccine* 28(6):1535–1540.

[Kempe, Kleinberg, and Tardos 2003] Kempe, D.; Kleinberg, J.; and Tardos, É. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 137–146. ACM.

[Les 2012] 2012. Network data. Website.

[McMahan, Gordon, and Blum 2003] McMahan, H.; Gordon, G.; and Blum, A. 2003. Planning in the presence of cost functions controlled by an adversary. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, volume 20, 536.

[Page et al. 1999] Page, L.; Brin, S.; Motwani, R.; and Winograd, T. 1999. The pagerank citation ranking: Bringing order to the web.

[Shetty 2010] Shetty, P. 2010. Experts concerned about vaccination backlash. *The Lancet* 375(9719):970–971.

[Smith 2010] Smith, T. 2010. A little birdie told me: H1n1 information and misinformation exchange on twitter.

[Trusov, Bucklin, and Pauwels 2009] Trusov, M.; Bucklin, R.; and Pauwels, K. 2009. Effects of word-of-mouth versus traditional marketing: Findings from an internet social networking site. *Journal of Marketing* 73(5):90–102.

[Tsai, Nguyen, and Tambe 2012] Tsai, J.; Nguyen, T.; and Tambe, M. 2012. Security games for controlling contagion.