Top-K Influential Nodes in Social Networks: A Game Perspective

Yu Zhang
Dept. of Computer Science
Peking University
Beijing, China
yutazh@pku.edu.cn

Yan Zhang
Dept. of Machine Intelligence
Peking University
Beijing, China
zhy@cis.pku.edu.cn

ABSTRACT

Influence maximization, the fundamental of viral marketing, aims to find top-K seed nodes maximizing influence spread under certain spreading models. In this paper, we study influence maximization from the perspective of game theory. We propose a Coordination Game (CG) model, in which every individuals make their decisions based on the benefit of coordination with their network neighbors, to study information propagation. Our model serves as the generalization of some existing models, such as Majority Vote model and Linear Threshold model. Under the generalized model, we study the hardness of influence maximization and the approximation guarantee of the greedy algorithm. We also combine several strategies to accelerate the algorithm. Experimental results show that after the acceleration, the accuracy of our algorithm is on par with the greedy algorithm's and better than other heuristics, while it is three orders of magnitude faster than the greedy algorithm.

CCS Concepts

•**Information systems** → **Data mining;** •**Theory of computation** → **Design and analysis of algorithms;**

Keywords

influence maximization; coordination game model; social networks; viral marketing

1. INTRODUCTION

Social networks play an important role in information diffusion. They give us motivation to use a small subset of influential individuals in a social network to activate a large number of people. Kempe et al. [9] build a theoretical framework of influence maximization, aiming to find top-K influential nodes under certain spreading models. They discuss two popular models - Independent Cascade (IC) model and Linear Threshold (LT) model and propose a greedy algorithm with $(1 - 1/e - \epsilon)$-approximation rate.

In most spreading models, each node has two states: active and inactive. Equivalently saying, it has two choices. We can model information diffusion as the process of individual decision-making. As individuals make their decisions based on choices of their neighbors, a particular pattern of behavior can begin to spread across the links of the network. Easley and Kleinberg [8] divide the cause of information propagation into two categories: information effects and direct-benefit effects. Obviously, IC model and LT model belong to the former one, while we focus on the latter one. In our Coordination Game (CG) model, every individuals make their decisions based on the benefit of coordination with their network neighbors.

Influence maximization under CG model is useful in viral marketing. Let us recall the example in [9]. A company would like to market a new product, hoping it will be adopted by a large fraction of the network. The company can initially target a few influential nodes by giving them free samples of the product. Then other nodes will probably switch to using the new product and give up the old product because of the following two reasons: (1) They have higher evaluation of the new product than the old one. (2) They have to coordinate with their neighbors because using different products brings a lot of inconvenience and reduces their benefits. (e.g., people using different operating systems may have some problems with compatibility when sharing file, and users from different kinds of social media platforms cannot communicate with each other timely.) Our model can describe these two reasons precisely.

In this paper, we study how to find Top-K influential nodes under CG model. Firstly, we propose our model which serves as the generalization of some famous spreading models, such as Majority Vote model [3] and the well-known Linear Threshold model [9]. We then prove some generalized theoretical results under CG model, including the NP-hardness of the optimization problem itself and the #P-hardness of computing the objective function. Then we try to find a good approximation algorithm for the problem. We embed our CG model into the scenario of general diffusion process defined by Mossel and Roch [12]. We prove that the objective function is monotone and submodular if and only if the *cumulative distribution function* (CDF) of people's threshold is concave, in which case the greedy algorithm can return a $(1 - 1/e - \epsilon)$ approximate solution.

As a traditional method, Kempe et al. [9] use 10,000 times of Monte Carlo simulations to approximate the objective function every time, but it takes us too much time on large-scale networks. To accelerate our algorithm, we use two efficient heuristics - LazyForward [10] and StaticGreedy [7] to reduce the times of simulations. Experimental results

Figure 1: Payoff matrix of the coordination game.

show that our `Greedy` and `Greedy++` algorithms can activate more nodes than other heuristics. Moreover, `Greedy++` runs faster than `Greedy` by three orders of magnitude.

Related Work. Kempe et al. [9] first build an algorithmic framework of influence maximization by transforming it into a discrete optimization problem. The approximation rate of their algorithm is guaranteed by the monotonicity and submodularity of the global objective function. Mossel and Roch [12] prove that the submodularity of the "local" influence function indicates the submodularity of the "global" objective function.

Morris [11] is the first to propose a coordination game model in contagion. Easley and Kleinberg use a whole chapter to give more extended discussions about the model in their textbook [8].

A lot of efforts have been made on efficient computing methods of the objective function. These work can be divided into two directions. On one direction, researchers try to reduce the number of trials that need Monte Carlo simulations, such as `CELF` [10]. On the other direction, researchers focus on how to calculate the influence spread efficiently. For example, Chen et al. [4, 5] use arborescences or DAGs to represent the original graph. Cheng et al. [7] propose a `StaticGreedy` strategy. They also introduce an interesting self-consistent ranking method in [6].

2. MODEL

In a social network $G = (V, E)$, we study a situation in which each node has a choice between two behaviors, labeled A and B. If nodes u and v are linked by an edge, then there is an incentive for them to have their behaviors match. We use a game model to describe this situation. There is a coordination game on each edge $(u, v) \in E$, in which players u and v both have two strategies A and B. The payoffs are defined as follows:

- if u and v both adopt strategy A, they will get payoffs $p_{uA} > 0$ and $p_{vA} > 0$ respectively;

- if they both adopt strategy B, they will get payoffs $p_{uB} > 0$ and $p_{vB} > 0$ respectively;

- if they adopt different strategies, they each get a payoff of 0.

The payoff matrix is shown in Figure 1.

We define the total payoff of player u as the sum of the payoffs it gets from all coordination games with its neighbors $N(u) = \{v | (u, v) \in E\}$. If u can get a higher total payoff when it adopt A than that when it adopt B, it will choose strategy A. Otherwise it will choose strategy B.

According to the actual situation, we have the following assumptions for the payoffs: (1) All p_{uA} and p_{uB} ($u \in V$) may not be equal to each other because each person in the social network values behaviors A and B differently. (2) p_{uA} and p_{uB} ($u \in V$) can either be constants or independent and identically distributed random variables because

the cascading behaviors in networks are always considered to have determinate principles with some stochastic factors.

Suppose u knows all the choices of its neighbors: there are x_B nodes adopting B and $x_A = deg(u) - x_B$ nodes adopting A. Obviously, u will adopt B if and only if

$$p_{uB} x_B \geq p_{uA} x_A = p_{uA}(deg(u) - x_B), \quad (1)$$

or equivalently,

$$x_B \geq \frac{p_{uA}}{p_{uA} + p_{uB}} deg(u) = \delta_u deg(u), \quad \delta_u \in [0, 1]. \quad (2)$$

Influence Maximization Problem. Suppose now the market is dominated by A (i.e., all nodes in the network choose A). Given a constant k, we want to find a seed set $S_0 \subseteq V$, $|S_0| \leq k$. Initially we let each node in S_0 adopt B (and they will never change their choices again). Time then runs forward in unit steps. In each step, each node decides whether to switch from strategy A to strategy B according to the payoff-maximization principle. We can regard the evolution of nodes' choices as a spreading process of B in the network. The spread of behavior B will finally stop in at most $n = |V|$ steps.

We define $S_i = |\{u \in V | u \text{ adopts } B \text{ in step } i\}|$, $i = 1, 2, ..., n$. Our objective function is (the expectation of) the nodes affected by B at the endpoint, or

$$\sigma(S_0) = \mathbb{E}_{\{p_{uA}, p_{uB} | u \in V\}}[|S_n|] = \mathbb{E}_{\{\delta_u | u \in V\}}[|S_n|]. \quad (3)$$

And our purpose is to maximize $\sigma(S_0)$ subject to $|S_0| \leq k$.

Our model can be regarded as the generalization of the following two famous spreading models.

Majority Vote Model. Suppose all p_{uA} ($u \in V$) are constants and are equal to each other. So are all p_{uB} ($u \in V$). Equivalently, let

$$p_A = p_{uA}, \quad p_B = p_{uB}, \quad \delta = \delta_u = \frac{p_A}{p_A + p_B}, \quad \forall u \in V. \quad (4)$$

δ is a constant threshold same to all nodes. When $p_A = p_B$, or $\delta = \frac{1}{2}$, the spreading model is called Majority Vote model, which is extensively studied in [3].

Linear Threshold Model. If we set $p_{uA} = 1$ and let p_{uB} follow a continuous power-law distribution, i.e., the probabilistic density function (PDF) of p_{uB} is

$$f_B(x) = \frac{\alpha}{(x+1)^\gamma}, \ x \geq 0, \ \gamma > 1, \ \alpha = \frac{1}{\int_0^\infty \frac{1}{(x+1)^\gamma} dx} = \gamma - 1, \quad (5)$$

then $\forall 0 \leq x \leq 1$,

$$\Pr[\delta_u \leq x] = \Pr[\frac{1}{1+p_{uB}} \leq x] = \Pr[p_{uB} \geq 1/x - 1]$$
$$= \int_{1/x-1}^{+\infty} f_B(t) dt = -(t+1)^{-\gamma+1} \Big|_{1/x-1}^{+\infty} = x^{\gamma-1}. \quad (6)$$

If we set $\gamma = 2$, then we will have $\delta_u \sim U[0, 1]$. This is the well-known Linear Threshold model where the weights on the edges adjacent to node u is $1/deg(u)$ (i.e., $b_{vu} = \frac{1}{deg(u)}, \forall u, v \in V$).

Hardness. Under CG model, we have the following hardness result.

THEOREM 1. *(1) Influence maximization under CG model is NP-hard. (2) Computing the objective function under CG model is #P-hard.*

PROOF. (1) Chen [3] proves the NP-hardness of Influence Maximization under Majority Vote model with $\delta = \frac{1}{2}$, which is enough to demonstrate the first result. We want to mention that Chen's result can be generalized to any $\delta \in [0, 1]$.

(2) Chen et al. [5] prove it is #P-hard to compute exact influence in general networks under LT model. They use the settings that $b_{vu} = const, \forall u, v \in V$ in their proof. We modify the proof and get the hardness result under our settings.[1] We reduce this problem from the problem of counting simple paths in a directed graph. Given a directed graph $G = (V, E)$, counting the total number of simple paths in G is #P-hard [14]. Let $n = |V|$ and $D = \max_{v \in V} deg_{in}(v)$. From G, we construct $n+1$ graphs $G_1, G_2, ..., G_{n+1}$. To get G_i $(1 \leq i \leq n+1)$, we first add $D + i - deg_{in}(v)$ "branching nodes" linking to node v for all $v \in V$. And then we add a node s linking to all nodes in V. Thus each node in G_i has $D + i + 1$ in-links except "branching nodes" and s.

According to our assumption, the weight on each edge in G_i is $w_i = \frac{1}{D+i+1}$. Let $S_0 = \{s\}$ and $\mathcal{P}$ denote the set of all simple paths starting from s in G_i. (Note that $\mathcal{P}$ is identical in all G_i because "branching nodes" are unreachable from s.) According to [5], we have

$$\sigma_{G_i}(S_0) = \sum_{\pi \in \mathcal{P}} \prod_{e \in \pi} w_i, \quad (1 \leq i \leq n+1), \tag{7}$$

where $\sigma_{G_i}(S_0)$ means $\sigma(S_0)$ in G_i. Let B_j be the set of simple paths of length j in $\mathcal{P}$ $(0 \leq j \leq n)$. We have

$$\sigma_{G_i}(S_0) = \sum_{j=0}^{n} \sum_{\pi \in B_j} \prod_{e \in \pi} w_i = \sum_{j=0}^{n} \sum_{\pi \in B_j} w_i^j = \sum_{j=0}^{n} w_i^j |B_j|. \tag{8}$$

We want to solve these $n+1$ linear equations with $n+1$ variables $|B_0|, |B_1|, ..., |B_n|$. Since the coefficient matrix is a Vandermonde matrix, $(|B_0|, |B_1|, ..., |B_n|)$ is unique and easy to compute.

Finally, we notice that for each $j = 1, 2, ..., n$, there is a one-to-one correspondence between paths in B_j and simple paths of length $j - 1$ in G. Therefore, $\sum_{j=1}^{n} |B_j|$ is the total number of simple paths in G. We complete our reduction. $\square$

3. ALGORITHMS

Submodularity. To find a greedy algorithm with approximation guarantee, the submodularity of the objective function is necessary.

We first recall the general diffusion process defined by Mossel and Roch in [12].

Suppose each node v in the social network $G = (V, E)$ has a threshold $\theta_v \sim U[0, 1]$ *i.i.d* and a "local" spreading function $f_v : 2^V \to [0, 1]$. Initially there is a seed set $S_0 \subseteq V$. After step $t \geq 1$,

$$S_t = S_{t-1} \cup \{v | v \in V - S_{t-1} \wedge f_v(S_{t-1}) \geq \theta_v\}. \tag{9}$$

The spreading process will stop in at most $n = |V|$ steps. So the objective function is $\sigma(S_0) = \mathbb{E}_{\{\theta_u | u \in V\}}[|S_n|]$.

We can also embed our model into the scenario of the general diffusion process.

Let F_δ be the cumulative distribution function of δ_u. Since $\delta_u \in [0, 1]$, we have $F_\delta(0) = 0$ and $F_\delta(1) = 1$. $\forall v$ and S, let

$$\theta_v = F_\delta(\delta_v), \quad f_v(S) = F_\delta(\frac{|S \cap N(v)|}{deg(v)}). \tag{10}$$

Suppose F_δ is continuous and strictly monotone increasing in $[0, 1]$, then F_δ^{-1} exists, and $\forall x \in [0, 1]$,

$$\Pr[F_\delta(\delta_v) \leq x] = \Pr[\delta_v \leq F_\delta^{-1}(x)] = F_\delta(F_\delta^{-1}(x)) = x. \tag{11}$$

So $F_\delta(\delta_v) \sim U[0, 1]$. Therefore

$$f_v(S) \geq \theta_v \iff F_\delta(\frac{|S \cap N(v)|}{deg(v)}) \geq \theta_v \iff$$
$$|S \cap N(v)| \geq F_\delta^{-1}(\theta_v) deg(v) \iff |S \cap N(v)| \geq \delta_v deg(v). \tag{12}$$

LEMMA 1. *Suppose F_δ is continuous and strictly monotone increasing in $[0, 1]$, f_v is monotone and submodular for any node v (in any graph)* **iff** *F_δ is concave in $[0, 1]$.*

PROOF. ($\Leftarrow$) If F_δ is concave in $[0, 1]$, let $g_v(S) = \frac{|S \cap N(v)|}{deg(v)}$. It is easy to prove that F_δ (or g_v) is monotone increasing and submodular (or concave). Therefore $f_v = F_\delta \circ g_v$ is also monotone and submodular.

($\Rightarrow$) If F_δ is not concave in $[0, 1]$, then $\exists a, b, \lambda \in [0, 1]$ such that

$$\lambda F_\delta(a) + (1 - \lambda) F_\delta(b) > F_\delta(\lambda a + (1 - \lambda)b). \tag{13}$$

Since F_δ is (uniformly) continuous and bounded, if we pick up three rational numbers $\frac{N_1}{M}, \frac{N_2}{M}$ and $\frac{p}{q}$ which are very close to a, b, λ respectively, we will have

$$\frac{p}{q} F_\delta(\frac{N_1}{M}) + \frac{q-p}{q} F_\delta(\frac{N_2}{M}) > F_\delta(\frac{N_1 p + N_2(q-p)}{Mq}) = F_\delta(\frac{N_3}{Mq}). \tag{14}$$

Let $X_i = (\frac{i}{Mq}, F_\delta(\frac{i}{Mq}))$ be the points on the curve of F_δ $(i = N_1 q, ..., N_2 q)$ and l_0 be the line across $X_{N_1 q}$ and $X_{N_2 q}$. We know that X_{N_3} is below l_0. Therefore $\exists K_1 \leq N_3 - 1$ and $K_2 \geq N_3$ such that
(1) X_{K_1} is above or in l_0 while X_{K_1+1} is below l_0.
(2) X_{K_2} is below l_0 while X_{K_2+1} is above or in l_0.
Let l_1 be the line across X_{K_1} and X_{K_1+1} and let l_2 be the line across X_{K_2} and X_{K_2+1}. We know that $k(l_1) < k(l_0) < k(l_2)$, where $k()$ is the slope of the line.

Assume there is a node v with Mq neighbors. Let S be the set of v's K_1 neighbors and T be the set of v's K_2 neighbors, where $S \subset T$. There is another neighbor $u \notin T$. Therefore

$$f_v(T \cup \{u\}) - f_v(T) = F_\delta(\frac{K_2 + 1}{Mq}) - F_\delta(\frac{K_2}{Mq}) = \frac{k(l_2)}{Mq}$$
$$> \frac{k(l_1)}{Mq} = F_\delta(\frac{K_1 + 1}{Mq}) - F_\delta(\frac{K_1}{Mq}) = f_v(S \cup \{u\}) - f_v(S), \tag{15}$$

which violates the submodularity of f_v. $\square$

It is not difficult for us to understand Lemma 1 intuitively because submodularity can be considered as a kind of concavity. F_δ being concave in $[0, 1]$ means that the distribution of people's threshold has a negative skewness in comparison with the uniform distribution, which means most people in the network are easy to be influenced, or they always have a higher evaluation on new products than old ones. This assumption is reasonable in some cases (e.g., the mobile phone market). F_δ being continuous and strictly monotone increasing in $[0, 1]$ is a technical assumption instead of an essential one. We define these two assumptions as the *concave threshold property*.

For the general diffusion process, Mossel and Roch [12] have proved that $\sigma(S_0)$ is monotone and submodular if and only if f_v is monotone and submodular for any $v \in V$. Using this result and Lemma 1, we can get Theorem 2 immediately.

THEOREM 2. *$\sigma(S_0)$ is monotone and submodular* **iff** *F_δ satisfies the concave threshold property.*

Theorem 2 provides a strong tool to judge the objective function's submodularity under certain spreading models. For example, under Majority Vote model, $\sigma(S_0)$ is not submodular because $F_\delta(x) = \mathbb{I}(x \geq \delta)$ is not concave in $[0, 1]$,

where $\mathbb{I}()$ is the indicator function. The counterexample is also easy to find according to the "non-concave point" $x = \delta$. In contrast, under Linear Threshold model, $\sigma(S_0)$ is submodular because $F_\delta(x) = x$ is concave in $[0, 1]$.

Up till now, we have proved the monotonicity and submodularity of the objective function under CG model with some necessary assumptions. Using the result in [9], the greedy algorithm given in Algorithm 1 (`Greedy`) returns a $(1 - 1/e - \epsilon)$-approximate solution. The algorithm simply selects seed nodes one by one, and each time it always selects the node that provides the largest marginal gain of the objective function. For the nonsubmodular case, recall that it is NP-hard to approximate the influence maximization problem within a factor of $n^{1-\epsilon}$ in general [9].

Speeding-Up Algorithm. Due to the hardness of computing $\sigma(S_0)$, we use two efficient heuristics - LazyForward [10] and StaticGreedy [7] to accelerate our algorithm.

We maintain a priority queue. When finding the next node, we go through the nodes in decreasing order of their marginal gain. If the marginal gain of the top node has not been updated, we recompute it, and insert it into the priority queue again. Instead of conducting a huge number of Monte Carlo simulations each time, we produce a rather small number of snapshots at the very beginning, and use the same set of snapshots in all iterations. We name the accelerated algorithm as `Greedy++`.

Algorithm 1 `Greedy`(k, σ)

1: initialize $S_0 = \emptyset$
2: **for** $i = 1$ to k **do**
3:     select $u = argmax_{v \in V - S_0}(\sigma(S_0 \cup \{v\}) - \sigma(S_0))$
4:     $S_0 = S_0 \cup \{u\}$
5: **end for**
6: output S_0

Algorithm 2 `Greedy++`(k, σ, R')

1: initialize $S_0 = \emptyset$
2: **for** $i = 1$ to R' **do**
3:     generate the threshold δ_v ($\forall v \in V$) for snapshot G_i
4: **end for**
5: **for all** $v \in V$ **do**
6:     $\Delta_v = +\infty$ //initialize the marginal gain of each node
7: **end for**
8: **for** $i = 1$ to k **do**
9:     **for all** $v \in V - S_0$ **do**
10:         $cur_v = false$
11:     **end for**
12:     **while** $true$ **do**
13:         $u = argmax_{v \in V - S_0}\Delta_v$ //maintain a priority queue
14:         **if** cur_u **then**
15:             $S_0 = S_0 \cup \{u\}$
16:             **break**
17:         **else**
18:             $\Delta_u = \frac{1}{R'}\sum_{i=1}^{R'}(\sigma_{G_i}(S_0 \cup \{u\}) - \sigma_{G_i}(S_0))$
19:             reinsert u into the priority queue and heapify
20:             $cur_u = true$
21:         **end if**
22:     **end while**
23: **end for**
24: output S_0

---

4. EXPERIMENTS

To test the effectiveness and efficiency of our `Greedy` and `Greedy++` algorithms, we conduct experiments on three real-world networks and compare our algorithms with other existing heuristics.

Datasets. The three real-world datasets include two collaboration networks `NetHEPT` [1] and `NetPHY` [1], and one online social network `Epinions` [2]. We summarize the statistical information of the these datasets in Table 1.

Datasets	$|V|$	$|E|$	Type
`NetHEPT`	15,233	58,991	Undirected
`NetPHY`	37,154	231,584	Undirected
`Epinions`	75,879	508,837	Directed

Table 1: Statistical information of three datasets.

Algorithms. A total of five algorithms are tested. Besides of `Greedy` and `Greedy++` proposed in this paper, we use other three heuristic algorithms as benchmark methods.[2]

(1) `PageRank` chooses nodes with the largest PageRank value. Since influential nodes are considered to have a large number of out-links, while nodes with high PageRank value are considered to have lots of in-links, we first change the direction of all edges in the graph and then run PageRank algorithm. We use $\alpha = 0.9$ as the random jump parameter. (2) `Degree` chooses nodes with the largest out-degree. (3) `Random` chooses nodes at random.

There are a lot of other efficient algorithms to solve influence maximization problem under IC model or LT model, such as `PMIA` [4], `LDAG` [5], and `IMM` [13]. But they cannot be applied in CG model directly, and we will not put them into the comparison.

Effectiveness. We first compare the effectiveness of `Greedy` and `Greedy++` with other algorithms by showing influence spread (i.e., $|S_n|$) of the obtained seed set.

In our CG model, distribution of δ_u can be various. We run influence maximization algorithms under four different spreading models in `NetHEPT`. In the four models, we set δ_u to be X, X^2, $\sqrt{X}$ and 0.5 respectively, where $X \sim U[0, 1]$. Therefore the distribution function $F_\delta(x)$ is x, $\sqrt{x}$, x^2 and $\mathbb{I}(x \geq 0.5)$ corresponding to the four cases.

According to Theorem 2, the first two cases are submodular cases, while the other two are not. Figure 2 shows our experimental results. In Figure 2, `Greedy++` consistently matches the performance of `Greedy` and significantly outperforms other heuristic algorithms in all cases. Because of the nonsubmodularity of the spreading models, the approximation rates of `Greedy` and `Greedy++` are not guaranteed in the third and the fourth cases. But our experimental results indicate that they still perform well in these cases. Besides, all the curves of `Greedy` and `Greedy++` are concave no matter in submodular or nonsubmodular cases. In two larger graphs `NetPHY` and `Epinions`, we get similar experimental results, which are shown in Figures 3(a) and 3(b).

We also compute the Jaccard similarity between the seed set selected by `Greedy` and that by other algorithms. The result is shown in Figures 3(c) and 3(d), from which we can see that `Greedy++` consistently shares much higher similarity with `Greedy` than `PageRank` and `Degree` do, meaning the introduction of LazyForward and StaticGreedy strategies will not cause significant loss of effectiveness.

Efficiency. We now test the running time of these algorithms. Figure 3 shows our experimental results.

As we expected, `Greedy++` runs consistently faster than `Greedy`, with more than three orders of magnitude speedup. For example, in the linear threshold case, it takes `Greedy`
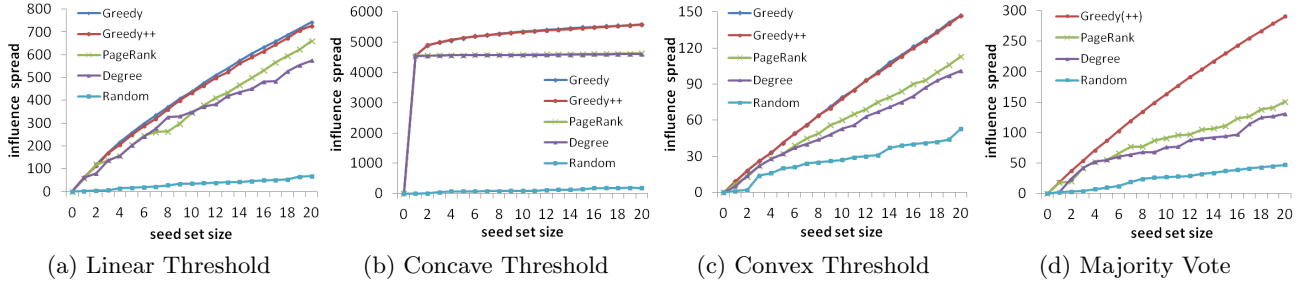
---

[2]The code and datasets can be found at `https://github.com/yuzhimanhua/Influence-Maximization`.

(a) Linear Threshold   (b) Concave Threshold   (c) Convex Threshold   (d) Majority Vote

Figure 2: Influence spread of various algorithms in `NetHEPT`, with different distribution of δ_u. ($X \sim U[0, 1]$.) (a) $\delta_u = X$ (submodular). (b) $\delta_u = X^2$ (submodular). (c) $\delta_u = \sqrt{X}$ (nonsubmodular). (d) $\delta_u = 0.5$ (nonsubmodular).
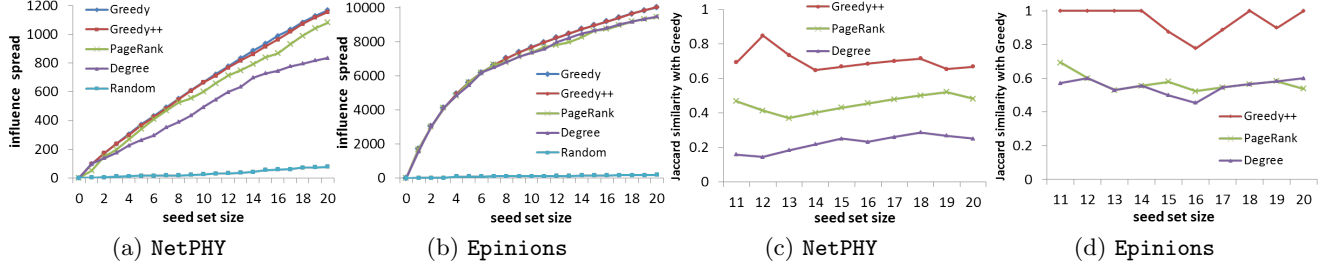

(a) `NetPHY`   (b) `Epinions`   (c) `NetPHY`   (d) `Epinions`

Figure 3: Influence spread of various algorithms in (a) `NetPHY` and (b) `Epinions`, and Jaccard similarity of the seed set with `Greedy` in (c) `NetPHY` and (d) `Epinions`. ($F_\delta(x) = x$.)

more than 9 days to get the top-20 influential nodes in `Epinions` while `Greedy++` only requires 8 minutes.

In other stochastic cases, our conclusion is still true. Note that `Greedy++` will spend more time if δ_u is small, or the influence spread tends to be wide. In Majority Vote model, the efficiency of the greedy algorithm dramatically rises because the estimation of influence spread becomes easy.
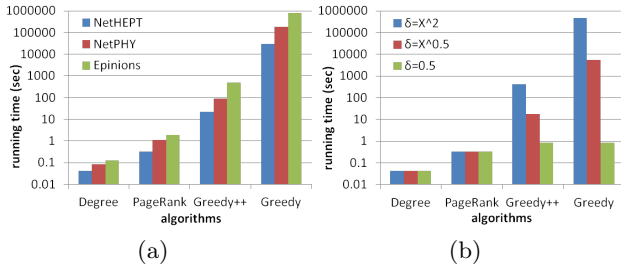

(a)   (b)

Figure 4: (a) Running time of various algorithms on three datasets. ($F_\delta(x) = x$.) (b) Running time of various algorithms in `NetHEPT`, with different distribution of δ_u. ($X \sim U[0, 1]$.)

5. CONCLUSIONS

In this paper, we have discussed how to find top-K influential nodes in social networks under a game theoretic model. We show the hardness of the optimization problem itself, as well as the hardness of calculating the objective function. We prove the approximation guarantee of the greedy algorithm under necessary assumptions. We also accelerate our algorithm with the combination of LazyForward and StaticGreedy. Our experimental results demonstrate that `Greedy++` matches `Greedy` in the spreading effect while significantly reducing running time, and it outperforms other heuristic algorithms such as MaxDegree and PageRank.

6. REFERENCES

[1] `http://research.microsoft.com/en-us/people/weic/graphdata.zip`.

[2] `http://snap.stanford.edu/data`.

[3] N. Chen. On the approximability of influence in social networks. In *SODA'09*, pages 1029–1037. ACM, 2009.

[4] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD'10*, pages 1029–1038. ACM, 2010.

[5] W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *ICDM'10*, pages 88–97. IEEE, 2010.

[6] S. Cheng, H. Shen, J. Huang, W. Chen, and X. Cheng. Imrank: Influence maximization via finding self-consistent ranking. In *SIGIR'14*, pages 475–484. ACM, 2014.

[7] S. Cheng, H. Shen, J. Huang, G. Zhang, and X. Cheng. Staticgreedy: solving the scalability-accuracy dilemma in influence maximization. In *CIKM'13*, pages 509–518. ACM, 2013.

[8] D. Easley and J. Kleinberg. *Networks, crowds, and markets: Reasoning about a highly connected world.* Cambridge University Press, 2010.

[9] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *KDD'03*, pages 137–146. ACM, 2003.

[10] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *KDD'07*, pages 420–429. ACM, 2007.

[11] S. Morris. Contagion. *The Review of Economic Studies*, 67:57–78, 2000.

[12] E. Mossel and S. Roch. Submodularity of influence in social networks: From local to global. *SIAM Journal on Computing*, 39(6):2176–2188, 2010.

[13] Y. Tang, Y. Shi, and X. Xiao. Influence maximization in near-linear time: a martingale approach. In *SIGMOD'15*, pages 1539–1554. ACM, 2015.

[14] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8:410–421, 1979.

APPENDIX

A. EQUILIBRIUM OF THE GAME

As a digression, we discuss our model from the perspective of game theory. An important problem is to find the *pure-strategy Nash equilibrium* (PSNE) of the game. In a PSNE, the strategy each node adopts is the best strategy toward its neighbors. Obviously, when all nodes adopt strategy A (or strategy B), the whole network will achieve a PSNE. But this is not meaningful. We want to know that given the initial state of each node, whether the network will converge to a PSNE or not. We take an early step in this problem, studying the case of Majority Vote model.

As is said before, the spread of a new behavior will finally stop in at most $|V|$ steps under CG model. But we have to point out that the final state is not a PSNE because nodes in the initial seed set may not choose the best strategy. To discuss the PSNE, we need to allow initial seed nodes to change their choices again. Therefore, the model can be explained as follows: Initially some nodes in the network choose strategy A, while others choose B. In each step, each node decides whether to change its strategy according to the payoff-maximization principle. Under this model, nodes can switch their decision for many times. And once all nodes stop changing their states, our game achieves a PSNE. To simplify to model, we assume that no one will meet a dilemma in the game (i.e., $p_B x_B \neq p_A x_A$ at any time[3]).

But will this equilibrium always appear? Not really. Let us consider a complete graph with 4 nodes (i.e., K_4). Initially two nodes choose A and the other two choose B, then all of the four nodes will "swing" between A and B forever. And the repeated game will become a "2-periodic" process, thus will never converge to a PSNE. Actually we have the following conclusion:

LEMMA 2. $\forall p_A, p_B \in \mathbb{Z}^+$, the repeated game will either go to a PSNE or become a "2-periodic" process in $O(\max\{p_A, p_B\}|E|)$ steps.

PROOF. We can assume that $p_A \geq p_B$. In step k, let $f_k(v) = p_A$ if node v adopts strategy A, and let $f_k(v) = -p_B$ if v adopts B. We define the potential function as

$$F_k = \sum_{(u,v) \in E} (f_k(u)f_{k-1}(v) + f_{k-1}(u)f_k(v))$$
$$= \sum_u \sum_{v \in N(u)} f_k(u)f_{k-1}(v) = \sum_u \sum_{v \in N(u)} f_k(v)f_{k-1}(u). \qquad (16)$$

Therefore

$$F_{k+1} - F_k = \sum_u \sum_{v \in N(u)} f_{k+1}(u)f_k(v) - \sum_u \sum_{v \in N(u)} f_k(v)f_{k-1}(u)$$
$$= \sum_u (f_{k+1}(u) - f_{k-1}(u))(\sum_{v \in N(u)} f_k(v)). \qquad (17)$$

Since $p_B x_B \neq p_A x_A$ at any time, we have $\sum_{v \in N(u)} f_k(v) \neq 0$. If $\sum_{v \in N(u)} f_k(v) > 0$, u should choose A in the next step and therefore $f_{k+1}(u) = p_A \geq f_{k-1}(u)$. If $\sum_{v \in N(u)} f_k(v) < 0$, u should choose B in the next step and therefore $f_{k+1}(u) = -p_B \leq f_{k-1}(u)$. In summary, $f_{k+1}(u) - f_{k-1}(u)$ always has the same sign with $\sum_{v \in N(u)} f_k(v)$. Therefore $F_{k+1} - F_k \geq 0$.

We can also prove that:
(1) $F_k \leq 2p_A^2|E|$, $F_k \geq -2p_A p_B |E|$.
(2) If $F_{k+1} - F_k \neq 0$, then $F_{k+1} - F_k \geq (p_A + p_B) \times 1$.

---

[3]e.g., $p_A = p_B = 1$ and each node has an odd number of neighbors.

So $\exists K \leq \frac{2p_A^2|E| - (-2p_A p_B |E|)}{p_A + p_B} = O(p_A|E|)$ such that $F_{K+1} - F_K = 0$, or

$$\sum_u (f_{K+1}(u) - f_{K-1}(u))(\sum_{v \in N(u)} f_K(v)) = 0. \qquad (18)$$

Since $\sum_{v \in N(u)} f_K(v) \neq 0$, we have $f_{K+1}(u) - f_{K-1}(u) = 0$ ($\forall u \in V$). So each node makes the same choice in step $K-1$ and step $K+1$. Therefore, they will make the same choice in step K and step $K+2$, and so on. The process then has a period of 1 or 2, corresponding to a PSNE or a "2-periodic" process respectively. □

With the help of Lemma 2, we can have an efficient algorithm for computing PSNE. We just simulate the evolution of each node's state. Once we find that the process becomes "2-periodic" (it only takes 2 more steps), we know that the network cannot achieve a PSNE. Otherwise we can get the PSNE in $O(\max\{p_A, p_B\}|E|)$ steps. The time complexity of the algorithm is $O(\max\{p_A, p_B\}|E|(|V| + |E|))$. We conclude the result in Theorem 3.

THEOREM 3. Suppose p_A, $p_B \in \mathbb{Z}^+$ and are fixed, given the initial state of each node, it is **polynomial-time** to answer the following questions: (1) Will the network converge to a PSNE? (2) If so, compute the PSNE.