An Efficient Linear Programming Based Method for the Influence Maximization Problem in Social Networks

Abstract

The influence maximization problem (IMP) aims to determine the most influential individuals within a social network. In this study first we develop a binary integer program that approximates the original problem by Monte Carlo sampling. Next, to solve IMP efficiently, we propose a linear programming relaxation based method with a provable worst case bound that converges to the current state-of-the-art $1 - 1/e$ bound asymptotically. Experimental analysis indicate that the new method is superior to the state-of-the-art in terms of solution quality and this is one of the few studies that provides approximate optimal solutions for certain real life social networks.

Keywords: Influence maximization, stochastic optimization, sample average approximation, pipage method

1. Introduction

With the vast penetration of online social networks in our lives, the distribution of ideas, information or new products is carried out through social networks more and more frequently than ever. As a result, influential individuals are of great importance as they are highly demanded by companies for viral marketing campaigns [11]. The influence maximization problem (IMP) aims to determine the most influential individuals within a social network under interest. Aside from its viral marketing applications, the same concept is studied in network security, computer virus detection, epidemics analysis, infrastructure planning, habitat conservation and wireless sensor networks [16, 15, 23, 3].

We define IMP as $\{\max \sigma(S) : |S| = k; S \subset \mathcal{V}\}$ over a stochastic network. Here the set of nodes is represented as V and the seed set S, which is a subset of $\mathcal{V}$ with a fixed size k, corresponds to the influential nodes. Finally, the objective function $\sigma(S)$ is a measure of the expected number of nodes influenced in the social network when a cascade is triggered by the initial seed set S.

1.1. Related Work

IMP has been attracting a great amount of interest in the last years and many researchers focus on different aspects of the problem [22]. The first model of IMP as a mathematical optimization problem is developed by Kempe et al. [5] (and its recent version [12]) where the problem is addressed as "Influence Maximization". They show that the objective function $\sigma(S)$ is submodular under two popular diffusion models: Independent Cascade (IC) and Linear Threshold (LT). Consequently, a naive greedy algorithm together with Monte Carlo simulations to estimate $\sigma(S)$ guarantees a $(1-1/e)$ approximation to the optimal solution of IMP [19]. The requirement of calculation of the influence function $\sigma(S)$ many times in the Monte Carlo simulation steps results in scalability issues, especially for large networks. Upon this observation, many improvements have been proposed for the Greedy method such as Cost Effective Lazy Forward (CELF) method [16]. For large networks, even the performance of CELF is still unsatisfactory and different approaches are applied to solve IMP in terms of computational efficiency rather than the solution quality. Among the many, PMC [21], EasySIM [6] and IMM [24] are currently some of the best performers while keeping the approximate bound guarantee. Recently, Wang et al. [25] present a novel bottom-k sketch based RIS framework (BKRIS) and Ko et al. [14] develop a hybrid method that combines path and community based IM to significantly accelerate the seed set selection procedure. For a detailed comparison of the state-of-the-art, the survey by Li et al. [17] provides the expected and worst-case complexity levels of many algorithms.

While most researchers focus on scalability of IMP, the interest in the solution quality is increasing as well. For certain applications of IMP, identifying the optimal seed set can be a priority, rather than scalability. For instance, in scenarios such as (i) determining the most crucial servers to protect in a mission-critical computer network, (ii) identifying the most influential people in a small candidate group of expensive celebrities or (iii) determining the most vulnerable patients for vaccination to avoid an epidemic, determining the optimal seed set is essential. Since IMP is proven to be NP-hard [5] obtaining optimal solutions for large instances is difficult within a reasonable duration. Nowadays, the optimization community has started to provide more contributions in the pursuit of developing efficient methods that provide optimal solutions. IMP is approximated to a combinatorial optimization problem by Sample Average Approximation (SAA) method in [15, 8] for obtaining approximate optimal solutions. Recently, Wu and Küçükyavuz [26] develop a delayed constraint generation algorithm to find the optimal solution to the sampled version of IMP, which is the current state-of-the-art.

1.2. Contribution and Outline

In this study, we focus on both the formulation and mathematical properties of IMP, so that an efficient method to obtain optimal solutions can be developed. Our first contribution is modeling the problem by using a simpler and much efficient discrete binary integer program than the one in our previous work [8], which is presented in Section 2.3. Our second contribution is a Linear Programming (LP) relaxation based method with a provable worst case bound, which is a direct consequence of this new formulation. For certain network instances, this method can solve IMP optimally in polynomial time (when the LP relaxation yields an integral solution). If a fractional solution emerges, an integer solution can be obtained in linear time with pipage rounding, resulting in a $(1-(1-1/p))^p - \epsilon)$ worst-case bound to the optimal solution. Here, p is the size of the largest predecessor set whose any member can activate a given node in the network and ϵ is the error term due to sampling. We prove that this bound is as good as the current state-of-the-art $(1-1/e)$.

The paper is organized as follows: In Section 2, the preliminaries of the IMP, its mathematical formulation and how it is solved by SAA is given. Section 3 presents the LP-relaxation based algorithm and derivation of its worst case bound. In Section 4, we present the computational results obtained by using some real life data. The last section gives the conclusion and future directions.

2. Basics of the Influence Maximization Problem

In this section, we first give a formal definition of our problem. Next, we introduce the Independent Cascade Diffusion model which describes how information propagates in the network. Following it, we explain how the exact influence of a seed set is determined on a running example and provide some hardness results. Then, we introduce our mathematical optimization model and its approximation by Monte Carlo sampling for practical use. Lastly, we describe how we solve the approximation version of our model using Sample Average Approximation framework. We conclude the section by providing certain mathematical relations that govern the quality of our approximations.

We define IMP over a weighted and directed network $\mathcal{N} = (\mathcal{V}, \mathcal{A}, \mathcal{W})$. In this network, $\mathcal{V}$ corresponds to the set of nodes that represent the members of the social network and its size is $|\mathcal{V}| = n$. The connections in the network such as being co-authors, followers, social media friends or similar relations are shown by arcs and the set of arcs is $\mathcal{A}$. Lastly, $\mathcal{W}$ is the set containing the weight information of these arcs. Each arc $(i, j) \in \mathcal{A}$ from node i to j has a weight of $p_{ij} \in \mathcal{W}$. It shows the probability of node i influencing node j whenever an action is triggered by node i in the social network. The size of set $\mathcal{A}$ is $|\mathcal{A}| = m$. For a given set of initial influentials (or seed set) S, which is the set of active nodes in the beginning of the cascading process, the expected number of activated or influenced nodes at the end of the process is calculated by the function $\sigma(S)$.

Given the network $\mathcal{N}$ and a diffusion model to describe how the information spreads over the network, we formally define IMP as identifying the optimal seed set whose size is k and when the diffusion process begins with the nodes in S, the influence function $\sigma(S)$ is maximized.

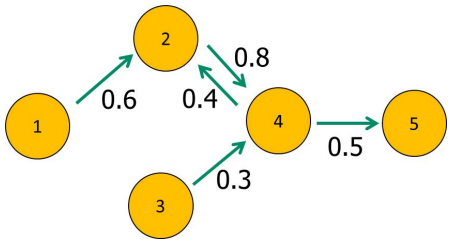


Figure 1: A sample social network with 5 nodes and 5 arcs

2.1. Independent Cascade Diffusion Model

It is assumed that the information propagates over the social network $\mathcal{N}$ obeying certain rules. To set these rules, some well-known diffusion models are used. In this study, independent cascade (IC) diffusion model is considered [7]. Regardless of the diffusion model, a node i is called as active or influenced if it does the action of interest or accepts the idea that is shared to it. Whenever a node becomes active, it can not deactivate itself and stays as an active node throughout the diffusion process. Under the IC diffusion model a node i that is activated in step t has only a single chance to successfully influence any neighbour j with a probability p_{ij}. These attempts are independent of each other and if node i is unsuccessful, it has no chance to influence node j any more. When a node i activates its neighbour j in step t, then node j tries to activate all of its inactive neighbours in step $t+1$. If there are more than one node trying to activate node j, they are sequenced arbitrarily. The process begins with an initial active seed set S and runs until no more activations are possible [3, 5].

A sample network with 5 nodes and 5 arcs with the influence probabilities on the arcs are displayed in Figure 1. To visualize the IC diffusion model, assume that node-1 is selected as the seed set. Then with probability 0.6 it will influence its only neighbour node-2. If node-1 is successful, in the next round node-2 will try to influence node-4 and so on.

2.2. Evaluation of Influence

The influence function $\sigma(S)$ is computationally difficult for exact calculation. Chen et al. [3] prove that it is #P-hard to compute $\sigma(S)$ precisely. $\sigma(S)$ is equivalent to the expected number of nodes that can be accessed by the elements of the seed set S of a corresponding random graph. Hu et al. [10] show that there are two possible methods for exact computation of $\sigma(S)$, similar to the computation of reliability or reachability both of which possess exponential-time complexity.

In the first method, all possible realizations (or scenarios) of the stochastic network are enumerated. Each realization is matched by a subset of active and inactive arcs in the network. We call an arc (i,j) as active if it exists in the network for a given realization. In the IC diffusion model the arc (i,j) is either active with probability p_{ij} or inactive with probability $(1-p_{ij})$. Since the number of arcs in the network is finite, the number of realizations is large but finite. Let $\mathcal{R}$ be the set of all possible realizations and r be the index of a given realization. Notice that $\mathcal{R}$ has an exponential size with $|\mathcal{R}| = 2^m$. Also, let μ_r be the probability of occurrence of realization r, which is computed by multiplying m probabilities, i.e. p_{ij} values for active arcs and $(1-p_{ij})$ values for the inactive arcs.

To see the intractability of exact calculation of $\sigma(S)$, we again refer to our example. In Figure 2, we display 3 of the possible $2^5 = 32$ scenarios. In the first scenario $r = 1$, we assume that all the arcs are active and the probability of this scenario is simply found by multiplying the influence probabilities ($\mu_1 = 0.0288$). In the second scenario $r = 2$, all the arcs except $(3,4)$ is active and in the last scenario $r = 32$ none of the arcs are active.

Also let's assume the seed set size to be $k = 2$. So since there are 5 nodes, there are $\binom{5}{2} = 10$ possible combinations for the seed set. We display only 3 of them as the rows of our example. Consequently, one has to compute the influence spread over each scenario and for each possible seed set to determine the exact expected influence of any possible seed set. For instance in Figure 2, in scenario $r = 1$, when the diffusion starts from the seed set {1,2}, a total of 4 nodes {1,2,4,5} are activated. It is simply calculated by counting

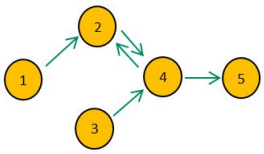
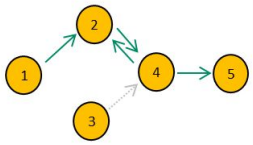
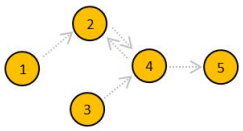
	r=1  μ_1=(0.6)(0.4)(0.8)(0.3)(0.5)=0.0288		r=2  μ_2=(0.6)(0.4)(0.8)(1-0.3)(0.5)=0.0672		...	r=32  μ_{32}=(1-0.6)(1-0.4)(1-0.8)(1-0.3)(1-0.5)= 0.0168		
Seed Set	Influence Spread	Contribution to σ(s)	Influence Spread	Contribution to σ(s)	...	Influence Spread	Contribution to σ(s)	$\sigma(s)$
{1,2}	4	0.1152	4	0.2688	...	2	0.0334	2.7492
{1,3}	5	0.1440	5	0.3360	...	2	0.0334	3.6184
...	...	...	...	...	...	...	...	...
{4,5}	3	0.0864	3	0.2016	...	2	0.0334	2.4086

Figure 2: Illustration of the Exact Calculation of $\sigma(S)$ Under IC Diffusion Model

all the accessible nodes via active arcs starting from the seed set. Therefore, the spread of {1,2} in scenario $r = 1$ is 4 and its contribution to the expected influence of {1,2} is simply the probability of scenario times the spread, i.e., $4 \times 0.0288 = 0.1152$. When all the influence contributions are summed over all the possible 32 scenarios, we can compute the exact expected influence for a given seed set. This is repeated for each seed set and the one yielding the maximum expected influence is determined as the optimal solution. The second method mentioned in [10] is a path-based method and its complexity is even higher than the first one, thus we prefer to skip the details of it.

2.3. Mathematical Formulation of IMP

As mentioned in the introduction, IMP can be formulated as $\{\max \sigma(S) : |S| = k; S \subset \mathcal{V}\}$. It is proven that IMP is NP-hard by showing a reduction to the Stochastic Set Covering problem [5]. Therefore, determining an optimal solution is very hard even for small networks. This form of IMP is inconvenient for mathematical optimization. thus, we benefit from the enumeration approach as we introduced for the exact computation of $\sigma(S)$.

In the proposed mathematical model, all nodes that can activate a given node i should be identified a priori. For this purpose, we define $\mathcal{P}_{ir} \subseteq \mathcal{V}$, which is the set of all predecessor nodes of i in the r-th realization. $\mathcal{P}_{ir}$ are constructed by running a breadth first search starting form node i and going in the reverse direction by using only the active arcs in a given realization. For instance, in Figure 2, consider node-4. In the first scenario $\mathcal{P}_{41} = \{1, 2, 3, 4\}$, because all these nodes can activate node 4 if they are selected as the seed set. In the second scenario $\mathcal{P}_{42} = \{1, 2, 4\}$, because arc $(3, 4)$ is not active any more. Lastly, when there are no incoming arcs to a node in a given scenario, $\mathcal{P}_{ir} = \{i\}$ (ex: $\mathcal{P}_{31} = \{3\}$).

We define two sets of decision variables, one for representing the seed set and the second to capture all the activated nodes for each scenario. First, let y be a $0 - 1$ vector of nodes in $\mathcal{V}$ and y_i is one if node i is selected as a seed and zero otherwise. Second, let $\mathcal{X}(y)$ be the set of random variables corresponding to all nodes that are activated in the diffusion process given the seed set y. Each $x_{ir} \in \mathcal{X}(y)$ is also a binary variable where x_{ir} is one if it is activated in the diffusion process in sample r or zero otherwise. The members of $\mathcal{X}(y)$ with value one are the ones that correspond to the seed nodes plus the nodes which are activated in the later steps of the IC diffusion process. Notice that, the probability distribution ruling the activation of the arcs (therefore the nodes connected to them) is independent of y. Given these definitions the objective function $\sigma(S)$ is rephrased as: $\sigma(S) = \sigma(y) = \sum_{i \in \mathcal{V}} E[\mathcal{X}(y)] = \sum_{r \in \mathcal{R}} \sum_{i \in \mathcal{V}} \mu_r x_{ir}$. Also, the function $\sigma(y)$ has the same meaning with $\sigma(S)$ but its domain is the set $\{0, 1\}^m$. All the sets, parameters, decision variables and other relevant terminology are summarized in Table 1 for quick reference.

Following the above notations and definitions, the Influence Maximization Binary Integer Program (IM-BIP) with independent cascade diffusion model is constructed as follows:

Table 1: Sets, parameters, decision variables and related notations

Sets	
$\mathcal{V}$	Set of nodes ($\lvert\mathcal{V}\rvert = n$)
$\mathcal{A}$	Set of arcs ($\lvert\mathcal{A}\rvert = m$)
$\mathcal{W}$	Set of arc weights
S	Seed set or initial set of influencers
$\mathcal{R}$	Set of all possible scenarios or realizations
$\mathcal{P}_{ir}$	Set of all predecessor nodes who can activate node i in scenario r
$\mathcal{R}_s$	Subset of scenarios obtained by Monte Carlo sampling
Parameters	
k	Seed set size
p_{ij}	Probability of node i to influence node j
μ_r	Probability of scenario r
R	number of possible scenarios or sample size
Decision Variables and Formulation Related Notation	
y_i	Binary decision variable to represent seed nodes
x_{ir}	Binary decision variable to represent if node i is activated in scenario r or not
z	Objective function value for IMBIP
z^*	Optimal objective function value of IMBIP
y^*	Optimal seed set (or optimal solution) of IMBIP
z_R	Objective function value for IMBIP-S
z_R^*	Optimal objective function value of IMBIP-S
y_R^*	Optimal seed set of IMBIP-S
Sample Average Approximation Related Notation	
M	Number of batches (integer programs) to be solved
R_j	Sample sizes used in step-$j = 1, 2, 3$ of SAA algorithm
$z_{iR_1}^*$	Optimal obj. funct. value of the i-th IMBIP-S in step-1
y_{iR}^*	Optimal seed set of the i-th IMBIP-S in step-1
$\hat{z}_{R_1}$	SAA obj. funct. value (upper bound estimate)
$z_{iR_2}^*$	Point estimate of $z_{iR_1}^*$ with a larger sample size in step-2
$\hat{y}_R^*$	Best seed set yielding the highest $z_{iR_2}^*$ value
$z_{R_3}^*$	Final point estimate for the seed set $\hat{y}_R^*$ (lower bound estimate)

$$\max \quad z = \sum_{r \in \mathcal{R}} \sum_{i \in \mathcal{V}} \mu_r x_{ir} \tag{1}$$

$$\text{s.t.} \sum_{i \in \mathcal{V}} y_i = k \tag{2}$$

$$x_{ir} \leq \sum_{j \in \mathcal{P}_{ir}} y_j \quad i \in \mathcal{V}, r \in \mathcal{R} \tag{3}$$

$$0 \leq x_{ir} \leq 1, y_i \in \{0,1\} \quad i \in \mathcal{V}, r \in \mathcal{R} \tag{4}$$

IMBIP aims to maximize the objective function (1) which calculates the expected number of activated nodes. The constraint (2) limits the size of the seed set to k, whereas, the constraints (3) capture the diffusion process. A node i in realization r is active only when it is in the seed set or it is connected to any member of the seed set through active arcs, which is captured by $\mathcal{P}_{ir}$. The formulation is completed with binary restrictions on y and non-negativity and unit bounds on x (4). Notice that, even though x_{ir} are initially defined as binary variables, the objective function and constraints (3) and (4) force them to be either 0 or 1, therefore we can relax the binary requirements. When solved optimally, IMBIP determines the optimal seed set y^* which attains the maximal influence value of z^*.

Sheldon et al. [23] developed a similar formulation, which is valid for only acyclic graphs. However, our formulation is superior as it can be used successfully for networks with cycles. For cyclic networks, when no precautions are taken in the formulation, a solution may contain some nodes becoming active without any of them being connected to any of the initial active nodes. Such solutions are unacceptable and it can be avoided by determining the predecessor set as we have done, or by using a less efficient formulation that requires an additional time index on the decision variables [8].

2.4. Solving IMP by Sample Average Approximation

Solving IMP (and also IMBIP) optimally is not practical as a consequence of the excessive number of possible realizations $|\mathcal{R}|$. Therefore, approximating $\sigma(S)$ using various sampling methods is a viable strategy. Among all, Monte Carlo sampling is used frequently to obtain a representative sub-network [5, 3, 24, 26]. For this purpose, a biased coin is flipped for each arc (i,j) with a success probability of p_{ij}. Thus, a sub-network $\mathcal{N}_r = (\mathcal{V}, \mathcal{A}_r)$ is constructed for each realization r, that contains only the active arcs according to the sampling outcome. Let $\mathcal{R}_s \subset \mathcal{R}$ denote the set of the realizations obtained by Monte Carlo sampling and let $\sigma_r(y)$ be the expected number of nodes reachable in sub-network $\mathcal{N}_r$ when the diffusion process is triggered by the seed vector y. By sampling we generate a total of $|\mathcal{R}_s| = R$ realizations and we define $z_R(y) = \frac{1}{R}\sum_{r=1}^{R} \sigma_r(y)$, which is a Monte Carlo Sample Average Approximation (SAA) of $\sigma(y) = \sigma(S)$ [15, 8]. Observe that $z_R(y)$ is an unbiased estimate of the original objective function $\sigma(S)$.

Then the SAA version of the optimization problem (IMBIP-S) becomes:

$$\max \quad z_R = \frac{1}{R} \sum_{r \in \mathcal{R}_s} \sum_{i \in \mathcal{V}} x_{ir} \tag{5}$$

$$\text{s.t.} \sum_{i \in \mathcal{V}} y_i = k \tag{6}$$

$$x_{ir} \leq \sum_{j \in \mathcal{P}_{ir}} y_j \quad i \in \mathcal{V}, r \in \mathcal{R}_s \tag{7}$$

$$0 \leq x_{ir} \leq 1, y_i \in \{0,1\} \quad i \in \mathcal{V}, r \in \mathcal{R}_s \tag{8}$$

Note that the major difference in this formulation is the new objective function and the much smaller realization set $\mathcal{R}_s$. Let y_R^* denote the optimal solution (seed set) to IMBIP-S. The fact that the number of all possible k combinations of y is finite (but obviously very large) guarantees that $\lim_{R\to\infty} z_R^* = \sigma(y_R^*)$ with probability one and similarly the limit points $\{y_R^*\}$ is optimal for IMBIP-S with probability one. Norkin et al. [20] proved that $\mathbf{E}[z_R^*] \geq \sigma(y_R^*)$ due to expectation relaxation. In other words, the optimal objective

function value of IMBIP-S is a positively biased estimator of the true optimum, even though the SAA function itself is not. Thus for a more accurate objective function estimate, a function evaluation, which is simply the point estimation of z^*_R is necessary. For this purpose, a much larger sample size $R' \gg R$ is used to compute $z_{R'} = \frac{1}{R'} \sum_{r \in \mathcal{R}'_s} \sum_{i \in \mathcal{V}} x_{ir}$ for the best SAA solution [4]. Notice that computation of $z_{R'}$ only requires the simulation of the diffusion for a fixed set of seed nodes, therefore it is very fast since it does not include any combinatorial optimization. To frame it up, the SAA procedure computes both an approximate upper bound (z^*_R) and a feasible (hopefully optimal) objective function estimate ($z^*_{R'}$) for the true optimum $\sigma(y^*)$. One can refer to [8] for the detailed implementation of SAA method that is used in this work, which is slightly different from [15]. However, for completeness we provide steps of the SAA method in Algorithm 1.

Algorithm 1 SAA Algorithm

1: Use M batches of samples of sample size R_1 for solving M integer programs to obtain the solutions y^*_{iR} and objective function values $z^*_{iR_1}, i = 1, ..., M$. Let $\hat{z}_{R_1} = \frac{1}{M} \sum_{i=1}^{M} z^*_{iR_1}$ and record it as an upper bound estimate to $\sigma(S)$.
2: Compute the influence values (point estimates) $z^*_{iR_2}(y^*_{iR})$ algorithmically, for the solutions $y^*_{iR}, i = 1, ..., M$ with a larger sample size R_2. Identify the solution with the best objective function value i.e., $\hat{y}^* = argmax_i\{z_{iR_2}(y^*_{iR})\}$ and record it as the optimal seed set estimate.
3: Compute $z^*_{R_3}(\hat{y}^*)$ algorithmically with a large sample size of $R_3 \geq R_2$ and record it as a lower bound estimate to $\sigma(S)$.

SAA is applied as a three step procedure. In the first step, M independent integer programs are solved which are constructed by M i.i.d batches of R_1 samples. Both the solutions and the objective function values are recorded. The average of these M solutions, $\hat{z}_{R_1}$ is recorded as an upper bound estimate to the original objective function. In the second step, the optimal solutions of each integer program of first step are compared with more accurate estimations. For this purpose, we algorithmically evaluate the objective function values, $z^*_{iR_2}(y^*_{iR})$, by using a much larger sample size R_2. The solution $\hat{y}^*$ yielding the highest objective function in step two is assumed to be the optimal solution (seed set) of the SAA method. In the final step of the SAA method, the objective function of the best solution, $\hat{y}^*$, is algorithmically re-computed with a new sample of size R_3, which is generally taken larger than or close to R_2. Finally, $\hat{y}^*$ and $z^*_{R_3}$ are reported as the estimates of the optimal seed set and optimal objective function value of the original problem. Notice that, since $\hat{y}^*$ is a feasible solution, $z^*_{R_3}$ is a lower bound (estimate) to the original problem. Therefore we have the approximate relation on the optimality gap: $\hat{z}_{R_1} \gtrsim \sigma(S) \gtrsim z^*_{R_3}$.

2.5. Quality of Approximations

A critical question arises about how to determine the sample size R (consequently R_1, R_2, R_3 and M) to obtain (i) a good (optimal) seed set and (ii) a good estimate of the original objective function σ. For the first case, we refer to the approximation results in Kleywegt et al. [13]. Let y^*_ϵ and $\hat{y}^*_\epsilon$ be the set of all ϵ-optimal solutions to IMBIP and IMBIP-S, respectively. By using the theory of large deviations (LD), they show that the probability of the event of having the optimal SAA solution ($\hat{y}^*_\epsilon$) as the true optimum solution (y^*_ϵ) converges to one exponentially fast as $R \to \infty$. By also benefiting from Cramér's LD theorem, they show that with a fixed significance level $\alpha \in (0, 1)$, the probability $P(\hat{y}^*_\epsilon \subset y^*_\epsilon) \geq 1 - \alpha$ i.e., any ρ-optimal solution of the SAA problem is an ϵ-optimal solution to the original problem with probability at least $1 - \alpha$, whenever the sample size,

$$R \geq \frac{3\sigma^2_{max}}{(\epsilon - \rho)^2} log\left(\frac{|\mathcal{X}|}{\alpha}\right) \tag{9}$$

where $\rho \in [0, \epsilon)$. In (9) $\sigma^2_{max} = max_{y \in \mathcal{X} \setminus y^*_\epsilon} Var[z(y^*) - z(y)]$ for an optimal solution $y^* \in y^*_\epsilon$ and $\mathcal{X}$ is the set of all possible feasible solutions to IMBIP-S [4]. Observe that the bound (9) may be too conservative for

practical use, resulting in excessively large integer programs. Nevertheless, it has interesting consequences for complexity issues. In (9) R depends only logarithmically both on the size of the feasible solution set $|\mathcal{X}|$ and on the tolerance probability α. An important implication of such behavior is the following: Even though the size of the feasible set $\mathcal{X}$ grows exponentially in the length of the problem input (number of nodes and sample size), the variance σ^2_{max} grows polynomially in the length of the problem input. Also the complexity of finding a ρ-optimal solution for the SAA problem grows polynomially in the length of the problem input. Then a solution can be generated in time that grows polynomially in the length of the problem input such that, with probability at least $1-\alpha$, the solution is ϵ-optimal to the original problem.

The computational complexity of solving the IMBIP-S as a stand-alone integer program grows exponentially in the number of decision variables and thus the sample size. Therefore, instead of solving a single IMBIP-S with a large sample size R, we prefer to use a smaller sample size R_1 and solve several IMBIP-S problems with i.i.d. samples. This approach is generally recommended in the SAA literature for better computational performance [13, 4, 18]. Each SAA problem can be thought as a Bernoulli trial with probability of success $p = p(R_1)$. Here success means that the optimal seed set identified by the SAA problem is the actual optimal seed set of the original problem. As mentioned above, this probability p tends to one as $R \to \infty$ exponentially fast. The probability of producing an optimal seed set of the true problem at least once in M trials is $1-(1-p)^M$ and this probability again tends to one exponentially as $M \to \infty$, given that $p > 0$. Therefore, instead of solving a single SAA problem with sample size MR_1, solving M separate SAA problems with sample size R_1 is preferable from the computational effort needed to solve integer programs. Given this conjecture, there is no guarantee that $p > 0$ for a given sample size R (or R_1). Particularly, for a finite R the probability p can be very small or even zero. Hence, even though this practical approach has empirical success, it does not contribute to the theoretical bound on the sample size R.

The choice of sample size at the second and third steps of the SAA method is similar to other IMP approaches. Classical Chernoff-Hoeffding bounds are valid as we algorithmically simulate the diffusion process for a fixed seed set y. Therefore, if the diffusion process is repeated for a given seed set y independently at least

$$R \geq \frac{n^2}{\epsilon^2} log\left(\frac{1}{\delta}\right) \tag{10}$$

times, then the average number of activated nodes over these runs is a $(1 \pm \epsilon)$ approximation to $\sigma(y)$, with probability at least $1-\delta$ [12].

As a summary, our first contribution is the new formulation of IMBIP and thus IMBIP-S. It is a simpler formulation compared to the one in [8], where the decision variables have an additional time index t showing the actual step when a node is activated in the diffusion process. In [8] breadth-first search procedure is run to find at which step a node is activated for each of its predecessors that can access to node i. Then the maximum of these values is recorded as T^{max}_{ir} for each node-sample couple. In that formulation, the predecessor set is not kept (although implicitly identified), but just the one-hop neighbours are kept as the neighbourhood set $\mathcal{N}_{ir}$ to construct the constraints. In the new formulation, we again apply the breadth-first search procedure and this time record the predecessor set $\mathcal{P}_{ir}$ for each node-sample couple. This process is similar to the reverse reachable set notion presented in [2] and the following studies. As a consequence, the time index can be dropped because it is not needed to construct the constraints any more. This approach significantly reduces the number of variables and constraints, thus larger instances of IMBIP-S can be solved even as stand-alone integer programs with sampling. More importantly, the new form of IMBIP-S has a special structure that facilitates the development of an LP-based solution technique, which is discussed in the next section.

3. An LP Relaxation based C-approximation algorithm

Scalability is a critical issue in influence maximization. Many real life social networks contain millions of nodes and up to billions of edges. As a result, most of the well-known studies in the literature are the ones that propose fast and scalable methods with relatively less emphasis on optimality. Nevertheless, these

algorithms try to keep their worst case bound to a reasonable level, mostly to $1 - 1/e$. However, in this work our objective is to find ϵ-optimal solutions and improve our scalability with minimum sacrifice from optimality. To achieve this, we benefit from the LP-relaxation of IMBIP-S.

In IMBIP-S, the seed set variables y_i dictate the optimal values for x_{ir} together with the unit bounds. Therefore, maximizing z_R is equivalent to maximizing the non-linear function $F(y) = \sum_{r \in \mathcal{R}_s} \sum_{i \in \mathcal{V}} \left(1 - \prod_{j \in \mathcal{P}_{ir}} (1 - y_j)\right)$ over all binary vectors y satisfying (6). Similarly, the objective function z_R can be replaced by the function $L(y) = \sum_{r \in \mathcal{R}_s} \sum_{i \in \mathcal{V}} min\{1, \sum_{j \in \mathcal{P}_{ir}} y_j\}$, which is favorable for an LP-relaxation based optimization scheme.

Ageev and Sviridenko [1] has shown that for the maximum k-coverage problem the following two conditions hold:

(i) The function $\psi(\epsilon, y, s, t) = F(y_1, .., y_s + \epsilon, ..., y_t - \epsilon, ...y_n)$ is convex with respect to $\epsilon \in [-min\{y_s, 1 - y_t\}, min\{1 - y_s, y_t\}]$ for each pair of indices s and t and each $y \in [0, 1]^n$,

(ii) There exists $C > 0$ such that $F(y) \geq CL(y)$ for each $y \in [0, 1]^n$.

Observe that IMBIP-S is a special case of the SAA version of stochastic maximum k-coverage problem. The objective function aims to find a ground set with size k that maximizes the average coverage for the given R instances of the network. The constraint (6) restricts the number of selected variables from the ground set to k and lastly the constraints (7) are equivalent coverage constraints. Different than the classical max-k-coverage problem, in IMBIP-S, the i-th constraint of (7) contains both x_{ir} and y_i at the same time for any r. So if a node is not connected to any other node, then $\mathcal{P}_{ir}$ contains only i, otherwise it contains i plus all the nodes that can access i in sample r. We will first show that these two conditions are also valid for IMBIP-S by benefiting from the arguments provided in [1].

Proposition 1. *For IMBIP-S, the function $\psi(\epsilon, y, s, t) = F(y_1, .., y_s + \epsilon, ..., y_t - \epsilon, ...y_n)$ is convex with respect to $\epsilon \in [-min\{y_s, 1 - y_t\}, min\{1 - y_s, y_t\}]$ for each pair of indices s and t and each $y \in [0, 1]^n$*

Proof. $\psi(\epsilon, y, s, t)$ will contain terms like $(1 - (1 - y_s - \epsilon)(1 - y_t + \epsilon) \prod_{j \in \mathcal{P}_{ir}/\{s,t\}} (1 - y_j)$ or will have y_s and y_t in separate products according to the network structure and the sampling outcome. Therefore the main coefficient of ϵ will always be either quadratic (if they are in the same product) or linear (if they are in separate products). Thus $\psi(\epsilon, y, s, t)$ is always convex for each pair of indices s and t and each $y \in [0, 1]^n$. □

Proposition 2. *There exists $C > 0$ such that $F(y) \geq CL(y)$ for each $y \in [0, 1]^n$, where $C = (1-(1 - 1/p)^p)$, and $p = max\{|\mathcal{P}_{ir}| : i \in \mathcal{V}, r \in \mathcal{R}_s\}$.*

Proof. Let $g(z) = 1 - (1 - z/p)^p$, where $z = min\{1, \sum_{j \in \mathcal{P}_{ir}} y_j\}$. By using the arithmetic-geometric mean inequality we can write:

$$1 - \prod_{j \in \mathcal{P}_{ir}} (1 - y_j) \geq 1 - (1 - z/p)^p$$

Observe that $g(z)$ is concave between $[0, 1]$ and $g(0) = 0, g(1) = 1 - (1 - 1/p)^p$, so $g(z) \geq (1 - (1 - 1/p)^p)\, z$. Therefore,

$$1 - \prod_{j \in \mathcal{P}_{ir}} (1 - y_j) \geq g(z) \geq (1 - (1 - 1/p)^p) \left(min\{1, \sum_{j \in \mathcal{P}_{ir}} y_j\}\right)$$

which completes the proof. □

Now, consider the LP-relaxation of IMBIP-S and its optimal solution y^*_{LP}. If y^*_{LP} is integral, then it is the optimal solution to its original integer version as well. Otherwise, there exists at least 2 fractional variables y_s and y_t due to (6). Of these two fractional variables, at least one of them can be made binary

by adding or subtracting one of the two possible values of ϵ, which is either $\epsilon = min(1 - y_s, y_t)$ or $\epsilon = -min(y_s, 1 - y_t)$. Since we are maximizing, the ϵ yielding the higher function value is preferred. This fractional-to-integer conversion of a fractional variable is called 'pipage'. Then, the new feasible solution $y'_{LP} = \{y_1, .., y_s + \epsilon, ..., y_t - \epsilon, ...y_n\}$ has less non-integer components. After repeating these 'pipage' steps at most $n - 1$ times, a binary feasible solution $\hat{y}_{LP}$ is obtained.

Theorem 1. *The pipage method is a C-approximation algorithm for IMBIP-S, where* $C = (1-(1-1/p)^p)$, *and* $p = max\{|\mathcal{P}_{ir}| : i \in \mathcal{V}, r \in \mathcal{R}_s\}$.

Proof. First, we will show that the worst case bound is asymptotically close to $C = (1 - (1 - 1/p)^p)$. Then we will explain how pipage method guarantees a feasible integer solution preserving the integrality gap.

Set $n = pk$. Let P_i be the collection of all subsets of nodes $\{1, 2, ...n\} - \{i\}$ with cardinality $p - 1$. We want to compute the lowest value for $z^*_R(IP)/z^*_R(LP)$ for IMBIP-S. Remember that the constraints (7) are constructed as $x_{ir} \leq y_i + \sum_{j \in \mathcal{P}_i/\{i\}} y_j$ with a total of p terms on the right-hand side, because the seed set variable y_i is always included in the constraint of x_{ir} for every sample. Then by symmetry any binary vector y with k units will provide a solution $z^*_R(IP) = k\binom{n-1}{p-1} + (n-k)\left(\binom{n-1}{p-1} - \binom{n-k-1}{p-1}\right)$. In this summation, the left term counts all x_{ir} having the same index with the elements of the seed set. Whereas, the right term is the expected influence of the remaining nodes that are activated later by the diffusion process, which occurs only when the predecessor set of x_{ir} contains at least one of the k seed nodes for the corresponding sample. For the LP relaxation, the vector with all $y_i = 1/p$ provides the optimal LP solution with $z^*_R(LP) = n\binom{n-1}{p-1}$, which means all nodes are activated in all samples. Now we can compute the ratio $z^*_R(IP)/z^*_R(LP)$ to provide an upper bound on the optimality gap C.

$$\begin{aligned}\frac{z^*_R(IP)}{z^*_R(LP)} &= \frac{k\binom{n-1}{p-1} + (n-k)\left(\binom{n-1}{p-1} - \binom{n-k-1}{p-1}\right)}{n\binom{n-1}{p-1}} \\ &= \frac{n\binom{n-1}{p-1} - (n-k)\binom{n-k-1}{p-1}}{n\binom{n-1}{p-1}}\end{aligned}$$

multiplying both the nominator and denominator with $1/p$ we get:

$$\frac{z^*_R(IP)}{z^*_R(LP)} = \frac{\binom{n}{p} - \binom{n-k}{p}}{\binom{n}{p}}$$

This is the same bound obtained in [1] so the proof follows:

$$\begin{aligned}\frac{z^*_R(IP)}{z^*_R(LP)} &= 1 - \frac{(n-k)!}{p!(n-k-p)!}\frac{p!(n-p)!}{n!} \\ &= 1 - \left(\frac{n-k}{n}\right)\left(\frac{n-k-1}{n-1}\right)...\left(\frac{n-k-p+1}{n-p+1}\right) \\ &\leq 1 - \left(\frac{n-k}{n}\right)\left(\frac{n-k-1}{n}\right)...\left(\frac{n-k-p+1}{n}\right) \\ &= 1 - \left(1-\frac{1}{p}\right)\left(1-\frac{1}{p}-\frac{1}{n}\right)\left(1-\frac{1}{p}-\frac{2}{n}\right)...\left(1-\frac{1}{p}-\frac{p+1}{n}\right) \\ &\leq 1 - \left(1 - \frac{1}{p} - \frac{p+1}{n}\right)^p\end{aligned}$$

and the last term tends to $(1 - (1 - 1/p)^p)$ when p is fixed and $n \to \infty$.

Assume that the LP-relaxation of IMBIP-S provides a fractional solution y^*_{LP}. After applying a pipage step to obtain the new solution y'_{LP} with less fractional variables, $F(y'_{LP}) \geq F(y_{LP})$ will always hold due to convexity of $\psi(\epsilon, y, s, t)$ for $\epsilon \in [-min\{y_s, 1 - y_t\}, min\{1 - y_s, y_t\}]$. When all the pipage steps are completed with no fractional variables left, a binary feasible solution $\hat{y}_{LP}$ with $F(\hat{y}_{LP}) \geq CL(y^*_{LP}) \geq CL(y^*) = CF(y^*)$ is obtained, where y^* is the actual optimal integer solution to IMBIP-S. □

The pipage method has a polynomial time complexity. It involves, first solving an LP relaxation of IMBIP-S, plus the additional pipage steps, which are simple function evaluations. Also its worst case bound

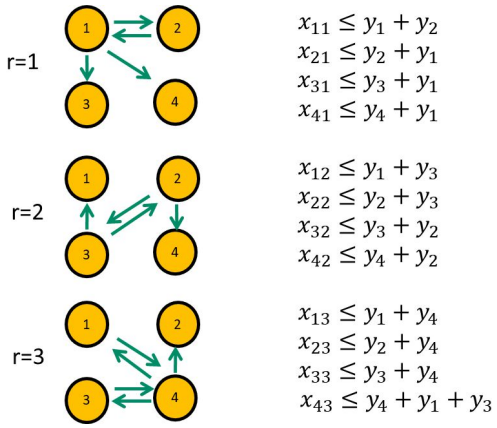


Figure 3: An example yielding a fractional LP solution

is as good as the greedy method. The greedy method solves IMBIP-S within a factor of $\left(1-(1-1/k)^k\right)$ [19]. For the asymptotic case our bound $(1-(1-1/p)^p)$ behaves similar to $\left(1-(1-1/k)^k\right)$, but for fixed p, the performance guarantee of the pipage method beats that of the greedy method. For instance, when the underlying network is bipartite, pipage method's bound strengthens to $3/4$, whereas the greedy method has still the $1-1/e$ bound [1].

To illustrate how the pipage method works consider Figure 3, with 4 nodes, 3 scenarios and $k=2$. Also the coverage constraints are displayed to show how the predecessor sets are translated into a mathematical form. For this instance, the optimal integer solution is $y_1^* = y_2^* = 1$ and $z^* = 3.66$. However the LP relaxation results in the fractional optimal solution with all $y_i = 0.5$ and $z_{LP} = 4$. Observe that, for this fractional solution y, $F(y) = 3.04$ and $L(y) = 4$. We randomly select two fractional variables, say y_1 and y_2. We determine two candidate ϵ values from $\epsilon = min(1-y_1, y_2)$ or $\epsilon = -min(y_1, 1-y_2)$. Since $y_1 = y_2 = 0.5$, $\epsilon = min(1-0.5, 0.5)$ or $\epsilon = -min(0.5, 1-0.5)$, then ϵ can be ± 0.5. When $\epsilon = -0.5$, the new solution is $y = \{0, 1, 0.5, 0.5\}$ with $F(y) = 3.17$ and $L(y) = 3.33$. When $\epsilon = +0.5$, the new solution is $y = \{1, 0, 0.5, 0.5\}$ with $F(y) = 3.25$ and $L(y) = 3.33$. Observe that $F(y)$ improves in both cases but we prefer $\epsilon = +0.5$ yielding a higher F(y) and the new solution becomes $y = \{1, 0, 0.5, 0.5\}$. We repeat these steps for y_3 and y_4 and obtain the final seed set of $y = \{1, 0, 1, 0\}$ or $y = \{1, 0, 0, 1\}$) with both solutions yielding $F(y) = 3.33$ and $L(y) = 3.33$. Since the final solution is integral we stop. Observe that the new integer solution to the LP relaxation is not optimal any more. In our example, if we don't select y_1 and y_2 together, pipage method ends up with the optimal solution $y^* = \{1, 1, 0, 0\}$ with $F(y^*) = 3.66$ and $L(y^*) = 3.66$. As a future research direction, it may be interesting to test various search methods to determine the best combinations of selecting the fractional couples to obtain the best possible integral solution.

We can easily integrate the pipage method to our solution approach. In the first step of Algorithm 1, instead of solving IMBIP-S as binary integer program, we solve its LP relaxation. If the solution is integer the algorithm proceeds just as it is. However, if the solution is fractional we obtain an integer solution by using the pipage method. So at the end of step one, we have M objective function values of the relaxed problems and M integer seed sets. These objective function values are used to compute the estimated upper bound $(z_{\hat{R}_1})_{LP} \geq (z_{\hat{R}_1})_{IP}$ and the corresponding seed sets are used in step-2 of the algorithm where the best seed set is identified by simulation as before. Basically, we lose from the tightness of our upper and lower bound estimates, but in return we guarantee a method with polynomial time complexity and potential for scalability. Overall the SAA method with pipage provides a $(1-(1-1/p))^p - \epsilon)$ approximation to IMBIP, where the ϵ is the approximation error that is carried to IMBIP-S from IMBIP due to sampling.

Table 2: The Summary of Data Sets

Data Set	Category	Nodes	Edges	Type
HEP arXiv	scale-free	37,153	231,568	Undirected
ego-Facebook	small world	4,039	88,234	Undirected
p2p-Gnutella4	peer-to-peer	10,876	39,994	Directed

4. Experimental Results

In this section we present our experimental results. First, we compare the performance of our new formulation provided in Section 2.3 with our old formulation in [8]. Next, we analyze the performance of our LP-based approach and compare both its solution quality and computational performance with the integer programming (IP) version and two other popular methods from the literature.

4.1. Experiment Setup and Data Sets

Three data sets are used in our experiments which are available in SNAP (https://snap.stanford.edu/data) and a brief summary of the networks are given in Table 2. The first one is **High Energy Physics (HEP) section of arXiv** data, which is also used in the experiments of [3, 5, 12, 8]. In this data set each node corresponds to an author and each arc is a co-authorship relation among authors. The co-authorship network displays a scale-free network characteristic, where a small number of nodes (authors) have high number of connections and most of the nodes have low number of connections [27, 8].

The second data set is **ego-Facebook**, which is a small world type network containing social circles (friend lists) from Facebook with $4,039$ nodes and $88,234$ arcs. The last data set is an internet peer-to-peer network example, **P2P-Gnutella04**, with $10,876$ nodes and $39,994$ arcs. A sequence of snapshots of the Gnutella peer-to-peer file sharing network from August 2002 are collected where nodes represent hosts in the Gnutella network topology and edges represent connections between the Gnutella hosts. This data set is also used in [26].

For the **arXiv** data set, we create 6 different sized networks with the number of arcs m={1K, 2K, 5K, 10K, 20K, 50K} that are selected randomly from the master dataset and 30 different seed set sizes, $k = 1, ..., 30$, meaning a total of 180 different test scenarios. For the **ego-Facebook** and **P2P-Gnutella04**, the complete data sets are considered and only seed set size is varied from $k = 1$ to $k = 30$. For the undirected networks, we assume the first node to be the tail and the second node to be the head. Placing two edges in both directions is a more common way of handling undirected networks, but since the networks are already large for the integer programs, we preferred such a setting for scalability. Since we are not interested in determining the actual best influential nodes of the given networks, but rather testing our approaches on random networks, this choice does not hurt our experimental analysis.

4.2. Methods Applied to Obtain Seed Sets

A total of four methods (IP, Pipage, CELF [16] and IMM [24]) are compared by testing their performance on the mentioned three data sets. To have a fair and accurate comparison of solution quality, we determine the best possible seed sets by each method with their default settings. Then the expected influence of these seed sets are computed algorithmically by simulating the diffusion process on the same batch of sampled networks with a large sample size.

The first method, IP, is the classical implementation of SAA as displayed in Algorithm 1, where we solve integer programs (IMBIP-S) at the first step. In method Pipage, we again use Algorithm 1, but in the first step we solve the LP relaxation of IMBIP-S and apply the pipage method as described in Section 3 to obtain a feasible seed set if the solution is fractional. In both IP and Pipage methods, the Monte Carlo SAA sample size $R_1 = 100$ and the number of SAA replications $M = 25$. So basically, $M = 25$ instances of the IP or LP are constructed with a sample size $R_1 = 100$ and then solved by using a commercial solver. For the point estimates (objective function evaluations) of the second step of the SAA procedure, the extended sample size is taken as $R_2 = 10,000$ [5, 8]. Notice that, since we need to identify the best seed set, we don't need

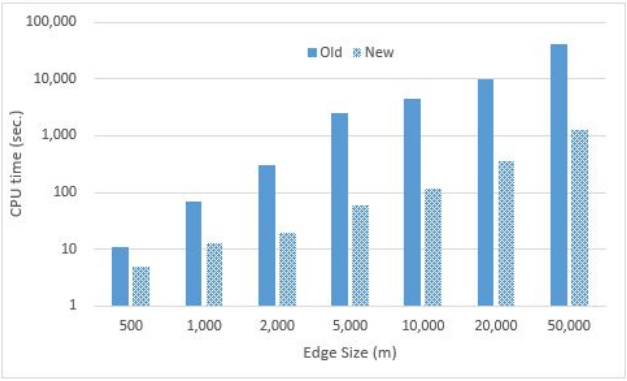


Figure 4: Running Time Comparison of BIP Formulations

to execute the step-3 of Algorithm 1. The choice of $M \in [20-30]$ is recommended in [4] to induce Central Limit Theorem for providing confidence intervals on the upper bound estimates. The choice of $R_1 = 100$ is an empirical preference, where the variance of the estimates start to stabilize around this level. We provide an extended analysis showing the quality of estimates with respect to varying values of R_1, which is also displayed in Figure 8.

The third method is the CELF implementation of the greedy method which is one of the fastest versions of it [16]. The sample size used in the greedy method is also $R = 10,000$ [5]. Last method is IMM - Influence Maximization with Martingales. It is one of the fastest algorithms to solve IMP with $1 - 1/e$ worst-case objective value guarantee. The IMM method is applied with its default parameter values where we set $\epsilon = 0.1$ [24].

The experiments are run on a server with two 64-bit, 3.00-GHz Xeon processors and 16GB RAM memory and the operating system is Windows 2008 R2 Server. GUROBI 7.5 is used for solving the IPs and LPs [9] with one hour time limit. If the problems are not solved within the time limit, for the upper bound we take the best bound given by Gurobi and for the seed set we take the best feasible solution in hand. Saying this, none of the IPs or LPs exceeded the time limit in our experiments.

4.3. Comparison of IMBIP Formulations

Our first analysis is on the efficiency of our new BIP formulation over [8]. The tests are carried out on random sub-graphs of **arXiv** data set with the number of arcs $m = \{1K, 2K, 5K, 10K, 20K, 50K\}$ and the seed set size $k = 1, ..., 30$. Algorithm 1 is applied with the aforementioned parameters. For a fair comparison, the instances generated with both formulations are run on the same hardware, with the default solver settings. The running times are recorded and averaged over all k values and displayed in Figure 4, i.e. the values displayed are averages of 30 results. Observe that the new formulation is 2 to 30 times faster than the old one, which is parallel to our expectations as the new formulation has much few number of variables and constraints. With this lighter formulation we can solve network instances with $m = 100,000$ from **arXiv** data set, which is twice as large as the largest instance reported in [8].

4.4. Solution Quality of the Seed Sets

In this sub-section, we make a comparison on the solution quality of the proposed methods. To have a fair comparison, the expected influence of each seed set obtained by each method are computed over the same sampled network with a large sample size $R = 10,000$. With this setting, if two methods have found the same seed sets, they will definitely produce the same expected influence, so that there will not be any deviation in objective function values due to sampling. On the contrary, when the seeds sets are different, we assume that the one yielding a larger influence is a better one with high probability and for very close values, our decision may be wrong but from a computational perspective the difference is assumed to be statistically insignificant.

Table 3: Expected Influence and Comparison of Methods on arXiv Data Set

		Optimality Gap (%)		
$n/m/k$	IP	Pipage	Greedy	IMM
12133/50000/30	970.88	0.12	0.71	2.20
12133/50000/20	823.39	0.98	0.56	2.68
12133/50000/10	577.65	0.15	1.39	1.52
12133/50000/5	373.95	0.12	1.14	2.50
5847/20000/30	533.34	0.37	0.99	2.33
5847/20000/20	434.79	0.24	0.8	3.01
5847/20000/10	304.94	0.29	-	0.27
5847/20000/5	198.56	0.05	-	0.08
3107/10000/30	319.92	0.06	-	1.61
3107/10000/20	274.33	-	0.01	1.12
3107/10000/10	205.51	0.11	-	2.78
3107/10000/5	151.04	0.15	-	1.66
1157/5000/30	201.03	0.13	0.75	2.71
1157/5000/20	171.78	-	0.28	3.68
1157/5000/10	139.36	0.03	0.04	1.08
1157/5000/5	120.70	-	-	2.04
378/2000/30	132.78	-	0.17	1.53
378/2000/20	112.05	0.15	-	1.46
378/2000/10	87.35	0.04	-	3.12
378/2000/5	71.99	-	-	3.52
378/1000/30	115.02	-	-	3.16
378/1000/20	95.15	0.17	-	2.31
378/1000/10	70.78	-	-	8.61
378/1000/5	55.99	-	-	9.02
	Average	0.13	0.30	2.67
	Std. Dev.	0.21	0.43	2.11

For **arXiv** data set, a representative sample of 24 out of 180 test scenarios is listed in Table 3. The first column shows the number of nodes and arcs of the sampled network and the seed set size in order. The second column shows the values of the point estimate of the best SAA solution ($z^*_{R_3}(\hat{y}^*)$), which is tagged as IP. This value is actually the best estimate for the true optimum objective function value $\sigma(S)$ in our setting. The last three columns display the percent gap between point estimates of IP and the point estimates of the pipage, greedy and IMM methods. The gap is calculated with the formula $\frac{a-b}{a} \times 100$, where a is $z^*_{R_3}(\hat{y}^*)$ and b is the point estimate value for the corresponding method. One can see that both the pipage and greedy method are performing quite well with very low gaps (0.13% and 0.30% on the average) and they find the same best solution in most of the cases. The gap for IMM is larger with 2.11% but again it is far from its theoretical worst case bound.

Table 4 displays the results for the **Facebook** data set. Again, Pipage and CELF are performing close to the best integer results with an average gap of 0.48% and 0.84%, respectively. IMM's performance is slightly worse than the others with an average of 2.98% gap. Observe that, **Facebook** network is denser than the other two networks, which increases the complexity of finding good seed sets and thus IP's performance superiority increases. Nevertheless, the differences between IP, Pipage and CELF are not statistically significant.

The results on our last data set **Gnutella** are somewhat controversial, which are presented in Table 5. Here IMM is the best performer, followed by Pipage, IP and CELF in order. This network is very sparse, hence the amount of sampling plays a significant role. As IMM method creates lots of samples compared to the other methods, it can identify the most influential seeds better than the others. To illustrate it better,

Table 4: Expected Influence and Comparison of Methods on Facebook Data Set

		Optimality Gap (%)		
k	IP	Pipage	Greedy	IMM
30	991.15	0.78	1.38	4.83
29	984.41	0.21	1.29	4.78
28	978.89	0.40	1.49	4.89
27	970.55	0.85	1.49	4.80
26	963.13	0.29	1.78	4.13
25	955.55	0.54	1.58	3.94
24	946.57	1.14	1.54	5.20
23	938.09	1.04	1.73	5.10
22	926.29	0.65	1.56	4.54
21	918.26	0.61	1.64	5.65
20	907.47	0.80	1.06	5.01
19	897.86	0.65	0.76	4.42
18	887.20	0.11	0.59	3.74
17	876.72	0.67	0.84	2.91
16	864.82	0.52	0.92	2.23
15	854.21	1.07	1.18	3.03
14	840.91	0.51	0.81	2.61
13	827.57	0.55	0.68	2.17
12	810.60	0.17	0.50	2.71
11	797.50	0.22	0.51	2.79
10	783.51	0.84	0.61	2.61
9	766.23	0.23	0.54	3.38
8	750.30	-	-	2.47
7	728.78	1.47	0.69	1.42
6	701.38	-	-	-
5	674.95	-	-	-
4	635.53	-	-	-
3	591.64	-	-	-
2	480.78	-	-	-
1	284.84	-	-	-
	Average	0.48	0.84	2.98
	Std. Dev.	0.40	0.61	1.85

Table 5: Expected Influence and Comparison of Methods on Gnutella Data Set

		Optimality Gap (%)		
k	IP	Pipage	Greedy	IMM
30	186.65	-0.09	0.24	-1.26
29	183.66	-0.06	0.91	-0.59
28	180.59	-	1.26	-0.30
27	176.53	-	1.12	-0.27
26	171.66	-0.04	0.39	-0.80
25	168.14	0.12	0.48	-0.12
24	163.65	-0.29	0.24	-0.55
23	160.67	-	0.59	0.29
22	155.83	-0.14	0.26	0.03
21	151.52	-0.09	0.14	-0.11
20	147.44	-	0.23	-0.06
19	143.73	0.06	0.23	0.07
18	139.19	0.03	0.18	0.16
17	134.71	-	0.16	0.03
16	129.91	-0.29	-0.29	0.03
15	125.34	-	0.25	0.25
14	120.18	-	-	-
13	115.00	-	-	-
12	108.99	-	-	0.03
11	103.06	-	-	0.12
10	96.85	-	-	-
9	90.36	-	0.04	-
8	83.87	-	0.03	-
7	77.55	-	-	0.45
6	70.75	-	-	-
5	62.87	-	-	-
4	52.22	-	-	-
3	41.09	-	-	0.18
2	29.98	0.02	-	-
1	18.82	-	-	-
	Average	-0.02	0.21	-0.07
	Std. Dev.	0.09	0.35	0.34

consider the instance of Gnutella network where $n = 10879, m = 39993, k = 30$. For this instance, IMM creates 247 reverse reachable set samples per node on the average, whereas this value is 100 in both IP and Pipage. In denser networks, the effect of under sampling is not as critical as in this sparse network.

4.5. Effects of LP Relaxation and Pipage Method on Upper and Lower Bound Estimates

As mentioned earlier, solving the LP relaxation of IMBIP-S in the SAA algorithm results in higher upper bound estimates. Similarly, rounding the fractional solutions by pipage method to get integral solutions may weaken the lower bound estimates. For the worst case, the degradation can be up to $1 - 1/e$ of the optimal solution. Table 6 provides useful insights to see the amount of degradation of the upper bounds. In Table 6, the second column displays the number of times the SAA upper bounds ($\hat{z}_{R_1}$) obtained by IP and LP Relaxation are not equal for $k = 1, .., 30$ cases. Remember that in each case, we solve $M = 25$ problems and average it to obtain $\hat{z}_{R_1}$. So the SAA objective function values of the two method will be equal only when all the 25 individual function values are exactly the same for both methods (i.e. yielding the same seed sets unless there is a multiple optima). Notice that, when the network gets larger or denser, the number of

Table 6: Comparison of solution quality

DataSet	# of times $\hat{z}_{LP} \neq \hat{z}_{IP}$	Avg. Gap (%)	Avg. # of pipage steps
arXiv (m=50K)	27	0.039	9.77
arXiv (m=20K)	14	0.010	4.19
arXiv (m=10K)	13	0.003	2.69
arXiv (m=5K)	24	0.024	9.06
arXiv (m=2K)	5	0.005	5.00
arXiv (m=1K)	15	0.018	8.00
Facebook	23	0.182	22.28
Gnutella	12	0.022	9.44

times we see a deviation increases. For smaller or sparse networks the bounds are identical most of the time, which means LP relaxation provides the same integer solution (or there is a multiple optima with a fractional solution having the same objective function value). The third column provides the average integrality gap between IP and LP objective function values. It is computed for only the cases with non-zero integrality gap. So for instance, in **arXiv** ($m = 50K$) case, 27 non-zero gap values are summed and divided by 27 to obtain 0.039% . The average gaps are very small and the largest difference occurs for the Facebook data set which is still smaller than 0.2%. Therefore we can conclude the LP relaxation of IMBIP-S is very strong with a very little integrality gap for the social networks under consideration. Theoretically LP relaxation can be as large as $e/(e-1)$ times the IP value, which is roughly 57%, but such a case is never observed. The last column shows the average number of pipage steps when a fractional solution is found. Again as the network gets denser and more connected the LP relaxation yields higher number of fractional y values and increases the number of pipage rounding steps.

One final insight is provided in Figure 5. It shows the relationship between average integrality gap and the seed set size. These are the average gap values computed over 8 data sets (arXiv(6), Facebook and Gnutella) for each k. For $k = 1$ the gap is zero and as the seed size increases, the gap gradually increases but even for $k = 30$ it is still lower than 0.1%. In [8], the budgeted version of IMP is solved by using a different formulation. In that work, the optimality gaps are insensitive to increasing budget, so that the gap values are not increasing or decreasing but rather zigzagging with increased budget (see Figure 3 in [8]). However, in our case, the increasing seed set size (in a sense budget) results in an increasing pattern of optimality gaps.

The comparison of lower bound estimates (thus the quality of seeds obtained by pipage method) is already available in Tables 3-5. Observe that, the gap between IP and Pipage methods' lower bound estimates are always very low. Out of a total 84 results listed, the highest difference observed is 1.47% (in the Facebook network). Therefore, we can assess that the strength of LP relaxation of IMBIP-S is reflected to the lower bound estimates as well.

4.6. Comparison of Running Times

We compare the computational performance of four methods and they are displayed in Figure 6. The values are the averages over $k = 1, ..., 30$ seed set scenarios. As expected IMM is very fast compared to the remaining three methods. In the slowest case it requires about 14 seconds to find the seed set. The running time of IP and Pipage are very close to each other and Pipage is slightly faster. Except a few cases greedy method is the slowest one and the gap increases as the data sets get larger and denser.

Figure 7 provides an interesting insight. It is obtained by averaging the computation times over 8 data sets for each k. It shows that our SAA based approach is quite insensitive to increasing seed set size k, however, greedy method's running time is almost linearly growing with respect to k. When k is small CELF is faster than SAA, but after a certain value of k (around $k = 5$ for large problems and $k = 15$ for small

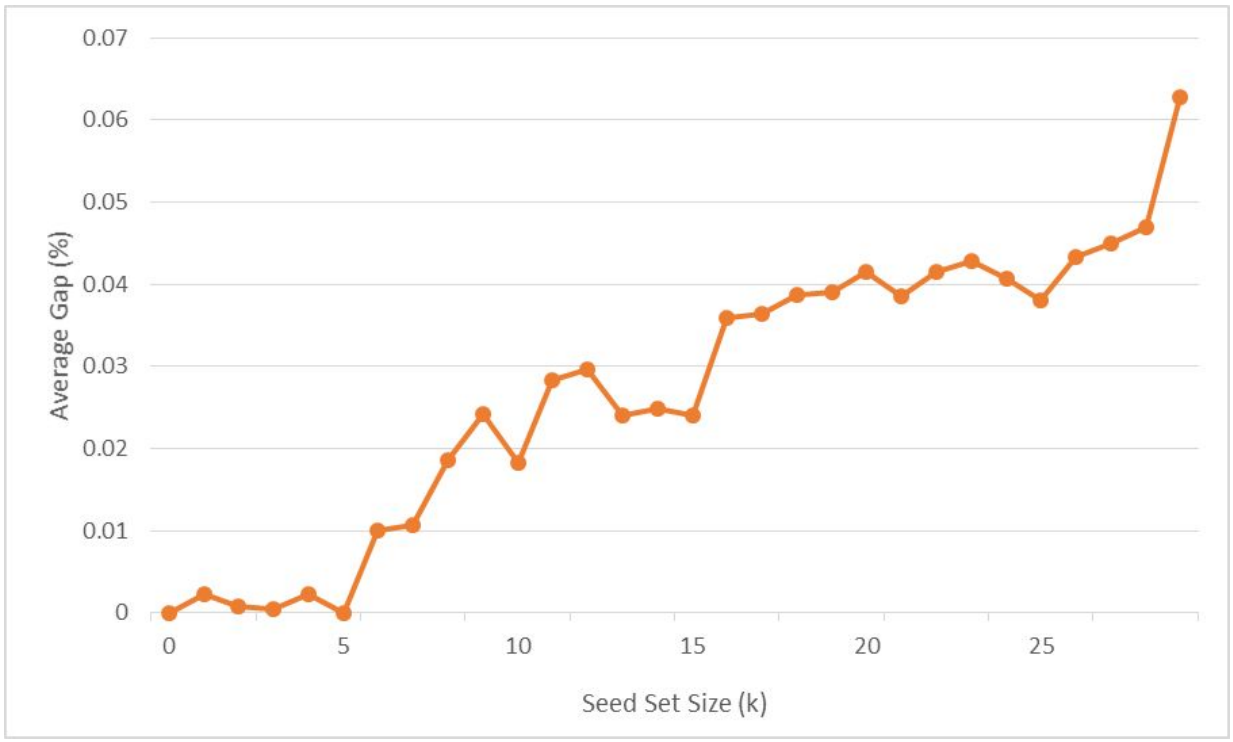


Figure 5: Relationship Between Seed Set Size and Integrality Gap

problems) our method becomes faster. Thus, especially for the instances with large seed set sizes our method is computationally favorable. Overall the contribution of our LP-based approach's computational efficiency is not very significant. This is mainly due to the fact that, the LP relaxation of IMBIP-S is strong and Gurobi Solver is solving the IP either in the root node or after a small number branch-and-bound steps. This phenomenon must be further investigated to see if there are some special structures in the problem that are exploitable using LP theory for better performance.

Lastly, we analyzed the effect of sample size (R_1) on the solution quality and running time. Figure 8 summarizes our findings. Here we tested different values of R_1 varying between 15 to 225 for fixed $k = 20$ on four different data sets (arXiv 10K, arXiv 50K, Facebook, GNutella). The left y-axis is displaying the standard deviation of the $M = 25$ upper bound values ($z^*_{iR_1}$) from their mean ($\hat{z}_{R_1}$). The running times are also significantly affected from increasing sample size, which are displayed as vertical bars (values displayed on the right y-axis with logarithmic running time scale).

We first analyze the line chart showing the standard deviation values of Facebook dataset (displayed with black dotted line) . There is a hinge at $R = 100$. Until that hinge, the standard deviation is decreasing consistently with increasing R, but after $R = 100$ the decrease is not significant any more. Therefore, one can conclude that $R = 100$ is a good choice for Facebook dataset, because increasing R further will only result in higher running times (resource consumption) with minor improvement in the quality of approximation. The hinge occurs at $R = 125$ for arXiv-L data set (sparsely dashed line), $R = 100$ for arXiv-S (straight line) and at around $R = 50$ for Gnutella dataset (tightly dashed line).

Next, we analyze the running times whose y-axis values (logarithmic scale) are on the right-hand side. Again we look for the R values where we observe a hinge. For T_{FB}(solution time of Facebook data set represented by the tallest bars) the hinge (even though it is not as clear as the std.dev case) is at $R = 75$. For the other 3 data sets we do not observe such a clear hinge and the height of the bars are growing almost linearly (in exponential scale). After all these observations, we conclude to take $R = 100$ in our tests, where we see the most number of hinges. However, this reasoning is subjective and for different purposes, the

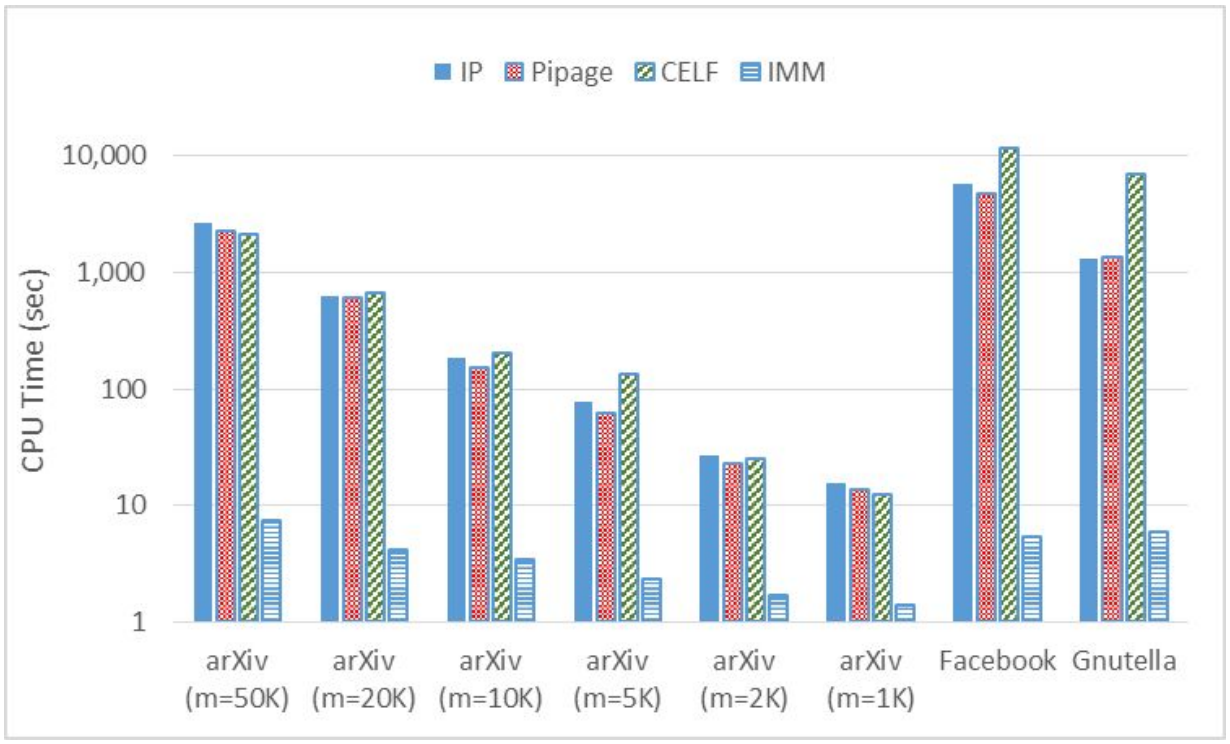


Figure 6: Comparison of Running Times

decision maker can prefer different R values. If one needs to solve IMP faster, smaller R values should be preferred and the risk of missing the best seed due to under sampling increases. Alternatively, one can use higher R values to increase the confidence on the seed set with much higher solution times. To summarize, the choice of sample size is critical to determine the trade-off between running time performance and stability of the algorithm for finding good seed sets.

The experimental analysis indicate that the proposed method is a significant step towards determining the optimal solution to IMP for small to mid-size social networks. One can determine the optimal seed set in polynomial time when the LP-relaxation yields an integral solution and if not, a close-to-optimal solution is easily achieved after applying pipage rounding to the fractional solution. Our methodology is generic and it can applied to many different types of social networks. Also the proposed method is still applicable for different diffusion models as long as the predecessor sets are obtained and translated into the constraints of our model. The computation of the predecessor sets requires a breadth-first-search for each node-sample couple, which results in scalability issues when the network is large and dense. The memory requirement of $\mathcal{O}(n^2R)$ or the computation time of $\mathcal{O}(n(m+n)R)$ is a significant overhead for large networks.

5. Conclusion

In this work we focused on the Influence Maximization Problem. We develop a binary-integer program to represent IMP with the Independent Cascade diffusion model. The original form of IMP is not favorable to be solved exactly, so we first converted it into a stochastic k-coverage problem and used the Sample Average Approximation (SAA) scheme to solve it. Then we proposed an LP-relaxation based pipage method with polynomial running time, which guarantees a $1 - 1/e - \epsilon$ worst-case optimality bound asymptotically. Experimental results over different networks and seed set values indicate that, the LP relaxation of IMP is very strong and the pipage method works very well and provides close-to-optimal solutions in most of the cases. Another important observation is the unexpectedly good performance of the Greedy and IMM

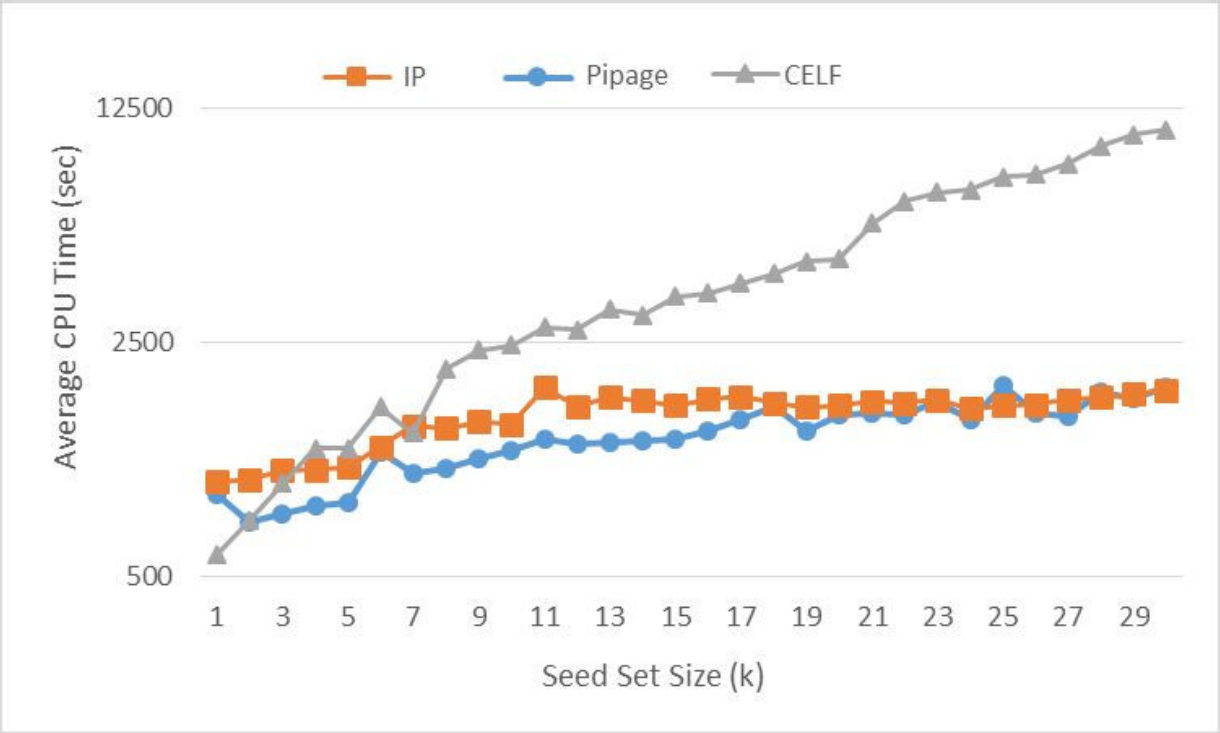


Figure 7: Running Time Performances With Respect to Seed Set Size

methods, where both methods provide close-to-optimal solutions. In terms of running time performance, the LP-relaxation based method is comparable with the greedy method and runs faster when the seed set size large. The superiority increases when the network size is larger and denser. As for the future research directions, the underlying LP structure can be analyzed and its sparsity may be exploited. Faster methods to solve the LP relaxation such as column generation or Lagrangean relaxation might be promising as well.

Acknowledgements

We would like to thank to the referees for pointing interesting directions and recommendations to improve the quality of our manuscript.

[1] Ageev, A., Sviridenko, M., Sep 2004. Pipage rounding: A new method of constructing algorithms with proven performance guarantee. Journal of Combinatorial Optimization 8 (3), 307–328.
URL https://doi.org/10.1023/B:JOCO.0000038913.96607.c2

[2] Borgs, C., Brautbar, M., Chayes, J., Lucier, B., 2014. Maximizing social influence in nearly optimal time. In: Proceedings of the Twenty-fifth Annual ACM-SIAM Symposium on Discrete Algorithms. SODA '14. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pp. 946–957.
URL http://dl.acm.org/citation.cfm?id=2634074.2634144

[3] Chen, W., Wang, C., Wang, Y., 2010. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In: Proc. of 16th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining. pp. 1029–1038.

[4] de Mello, T. H., Bayraksan, G., 2014. Monte carlo sampling-based methods for stochastic optimization. Surveys in Operations Research and Management Science 19 (1), 56 – 85.

[5] D.Kempe, Kleinberg, J., E.Tardos, 2003. Maximizing the spread of influence through a social network. In: Proc. of 9th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining. pp. 137–146.

[6] Galhotra, S., Arora, A., Roy, S., 2016. Holistic influence maximization: Combining scalability and efficiency with opinion-aware models. In: Proceedings of the 2016 International Conference on Management of Data. SIGMOD '16. ACM, New York, NY, USA, pp. 743–758.
URL http://doi.acm.org/10.1145/2882903.2882929

[7] Goldenberg, J., Libai, B., Muller, E., 2001. Talk of the network: A complex systems look at the underlying process of word-of-mouth. Marketing Letters, 211–223.

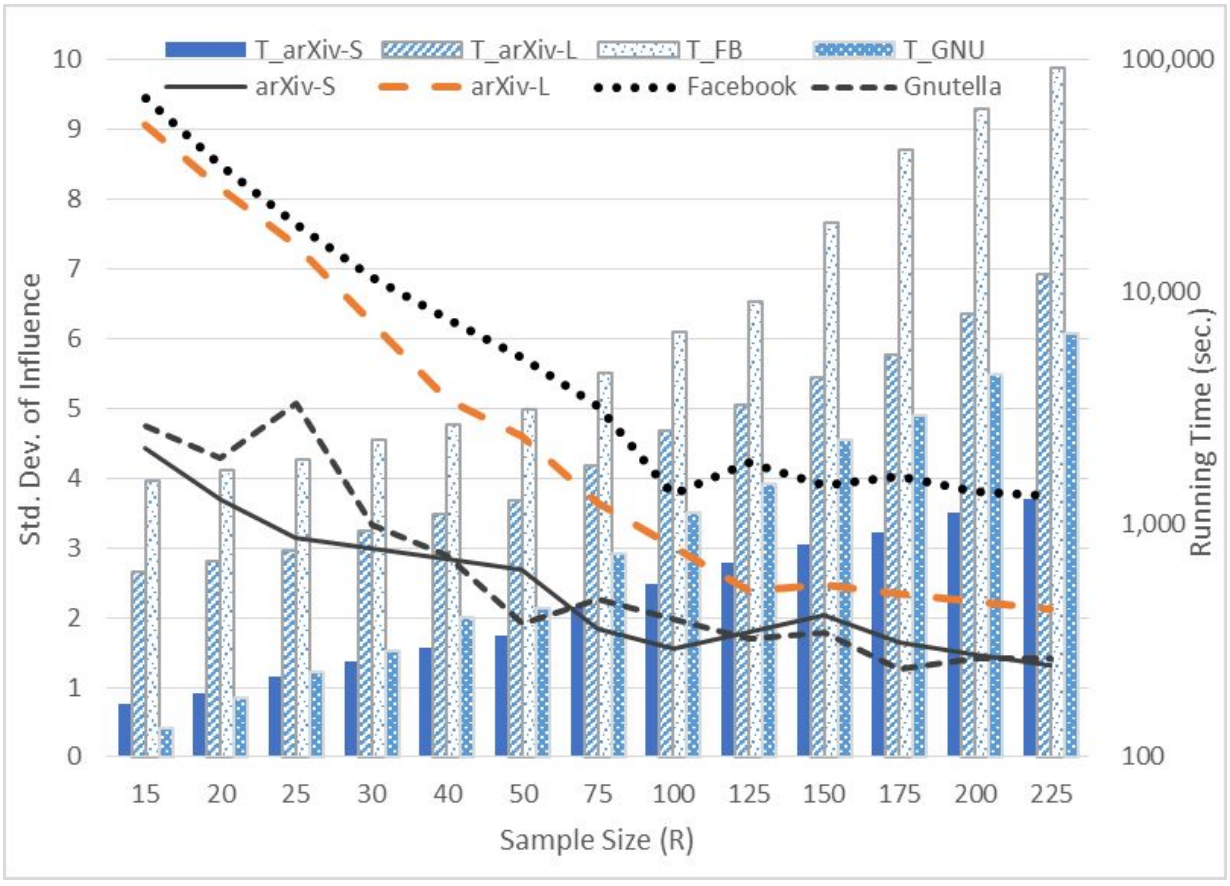


Figure 8: Effect of Sample Size on Upper Bound Estimates and Running Time

[8] Güney, E., 2017. On the optimal solution of budgeted influence maximization problem in social networks. Operational Research.
URL http://dx.doi.org/10.1007/s12351-017-0305-x
[9] Gurobi 7.5 User's Manual, 2017. Gurobi 7.5 User's Manual. Gurobi Optimization.
[10] Hu, J., Meng, K., Chen, X., C.Lin, Huang, J., 2014. Analysis of influence maximization in large-scale social networks. SIGMETRICS Perform. Eval. Rev. 41 (4), 78–81.
URL http://doi.acm.org/10.1145/2627534.2627559
[11] Iyengar, R., den Bulte, C. V., Valente, T., 2011. Opinion leadership and social contagion in new product diffusion. Marketing Science 30 (2), 195–212.
[12] Kempe, D., Kleinberg, J., Tardos, E., 2015. Maximizing the spread of influence through a social network. Theory of Computing 11 (4), 105–147.
URL http://www.theoryofcomputing.org/articles/v011a004
[13] Kleywegt, A. J., Shapiro, A., Homem-de Mello, T., Feb. 2002. The sample average approximation method for stochastic discrete optimization. SIAM J. on Optimization 12 (2), 479–502.
URL https://doi.org/10.1137/S1052623499363220
[14] Ko, Y.-Y., Cho, K.-J., Kim, S.-W., 2018. Efficient and effective influence maximization in social networks: A hybrid-approach. Information Sciences 465, 144 – 161.
[15] Lee, J., Hasenbein, J., Morton, D., 2015. Optimization of stochastic virus detection in contact networks. Operations Research Letters 43, 59–64.
[16] Leskovec, J., Krause, K., Guestrin, C., Faloustos, C., VanBriesen, J., Glance, N., 2007. Cost-effective outbreak detection in networks. In: Proc. of 13th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining. pp. 420–429.
[17] Li, Y., Fan, J., Wang, Y., Tan, K. L., 2018. Influence maximization on social graphs: A survey. IEEE Transactions on Knowledge and Data Engineering, 1–1.
[18] Linderoth, J., Shapiro, A., Wright, S., 2006. The empirical behavior of sampling methods for stochastic programming. Annals of Operations Research 142 (1), 215–241.
URL http://dx.doi.org/10.1007/s10479-006-6169-8
[19] Nemhauser, G., Wolsey, L., Fisher, M., 1978. An analysis of approximations for maximizing submodular set functions. Mathematical Programming 14, 265–294.
[20] Norkin, V., Pflug, G., Ruszczyski, A., 1998. A branch and bound method for stochastic global optimization. Mathematical Programming 83 (1-3), 425–450.
URL http://dx.doi.org/10.1007/BF02680569
[21] Ohsaka, N., Akiba, T., Yoshida, Y., Kawarabayashi, K.-I., 2014. Fast and accurate influence maximization on large networks with pruned monte-carlo simulations. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence. AAAI'14. AAAI Press, pp. 138–144.
URL http://dl.acm.org/citation.cfm?id=2893873.2893897
[22] Rabade, R., Mishra, N., Sharma, S., 2014. Survey of influential user identification techniques in online social networks. In: In: Thampi S., Abraham A., Pal S., Rodriguez J. (eds) Recent Advances in Intelligent Informatics. Advances in Intelligent Systems and Computing. pp. 359–370.
[23] Sheldon, D., Dilkina, B., Elmachtoub, A., Finseth, R., Sabharwal, A., Conrad, J., Shmoys, D., Allen, W., Amundsen, O., Vaughan, B., 2010. Maximizing the spread of cascades using network design. In: Proc. of the 26th Conf. on.Uncertainty in Artificial Intelligence. pp. 517–526.
[24] Tang, Y., Shi, Y., Xiao, X., 2015. Influence maximization in near-linear time: A martingale approach. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. SIGMOD '15. ACM, New York, NY, USA, pp. 1539–1554.
URL http://doi.acm.org/10.1145/2723372.2723734
[25] Wang, X., Zhang, Y., Zhang, W., Lin, X., Chen, C., 2017. Bring order into the samples: A novel scalable method for influence maximization. IEEE Transactions on Knowledge and Data Engineering 29 (2), 243–256.
[26] Wu, H.-H., Küçükyavuz, S., Oct 2017. A two-stage stochastic programming approach for influence maximization in social networks. Computational Optimization and Applications.
URL https://doi.org/10.1007/s10589-017-9958-x
[27] Yan, E., Ding, Y., 2009. Applying centrality measures to impact analysis: A coauthorship network analysis. Jour. of the Assoc. for Information Science and Technology 60 (10), 2107–2118.
URL http://dx.doi.org/10.1002/asi.v60:10