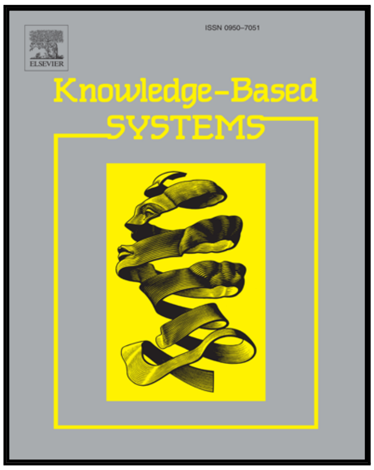

Influence Maximization in Social Networks under Deterministic Linear Threshold Model

Furkan Gürsoy, Dilek Günneç

PII: S0950-7051(18)30390-3
DOI: 10.1016/j.knosys.2018.07.040
Reference: KNOSYS 4440

To appear in: *Knowledge-Based Systems*

Received date: 8 December 2017
Revised date: 26 July 2018
Accepted date: 28 July 2018

Please cite this article as: Furkan Gürsoy, Dilek Günneç, Influence Maximization in Social Networks under Deterministic Linear Threshold Model, *Knowledge-Based Systems* (2018), doi: 10.1016/j.knosys.2018.07.040

Influence Maximization in Social Networks under Deterministic Linear Threshold Model

Furkan Gürsoy

Dept. of Management Information Systems, Boğaziçi University, 34342, Istanbul, Turkey

Dilek Günneç*

Dept. of Industrial Engineering, Özyeğin University, 34794, Istanbul, Turkey

Abstract

We define the new Targeted and Budgeted Influence Maximization under Deterministic Linear Threshold Model problem and develop the novel and scalable TArgeted and BUdgeted Potential Greedy (TABU-PG) algorithm which allows for optional methods to solve this problem. It is an iterative and greedy algorithm that relies on investing in potential future gains when choosing seed nodes. We suggest new real-world mimicking techniques for generating influence weights, thresholds, profits, and costs. Extensive computational experiments on four real network (Epinions, Academia, Pokec and Inploid) show that our proposed heuristics perform significantly better than benchmarks. We equip TABU-PG with novel scalability methods which reduce runtime by limiting the seed node candidate pool, or by selecting more nodes at once, trading-off with spread performance.

Keywords: Influence Maximization, Social Networks, Diffusion Models, Targeted Marketing, Greedy Algorithm

*Corresponding author

Email addresses: furkan.gursoy@boun.edu.tr (Furkan Gürsoy), dilek.gunnec@ozyegin.edu.tr (Dilek Günneç)

1. Introduction

Social process of influence intensely and frequently takes place among people. As a result, people's decisions and behaviours are influenced by others. Such influence can be observed in marketing, consumer behavior, politics, persuasion, peer pressure, conformity, and leadership.

Furthermore, influence may occur at a conscious level or at an unconscious level. At a conscious level, people choose whether to be influenced or not as a result of some rational decision making process. There are two types of benefits in imitating the decisions of others: the direct-benefit effect and the informational effect. Direct-benefit effect takes place when one's payoff from her own action is directly affected by other people's actions. This phenomenon is also called as network effect. It can be illustrated by the historical adoption rates of fax machine, which quickly peaked after it slowly reached a tipping point [1]. Informational effect occurs when collective information is more powerful than one's private information. In a setting where one has limited information on which action to prefer, the decision is more likely to be made by mimicking others' decisions. On the other hand, one does not always have the control over whether to be influenced or not as influence can also happen at an unconscious level. Social conformity, mirroring and other psychological effects are examples of such situations [2].

The earliest studies about influence propagation in social networks took place in the middle of the 20$^{\text{th}}$ century. In his famous book Rogers [3] brings together a number of studies which study how innovations in agricultural methods and tools spread in the rural communities. He paved the way for the development of notions such as strength of weak ties, tipping point, phases of adoption, and categories of technology adopters.

Given the importance of and opportunities in social influence, marketers try to take advantage of it in order to increase their market recognition and adoption of their products. For companies, a well-planned, calculated and targeted viral marketing in the form of an "influencer marketing campaign" can trigger a cascading positive word-of-mouth effect [4]. Ideally, subsidizing a few influential people to promote a certain brand will create a cascade in the network. Therefore, the problem is to select a set of influentials in such a way where influence spread is maximized while the cost of subsidizing the influentials is kept within a given budget.

As a motivating example, consider an online baby products retailer who wants to advertise over a social network. The main target market for this company is the people who have or expect to have babies. Such people might be characterized by age groups or online behaviour. The target market is further segmented to subgroups, for example with respect to income levels, which carry different customer lifetime values (e.g., expected profit) for the retailer. The retailer sets a budget to promote its products to its target market via influencers who possess varying degrees of self-perceived values and impact in the social network and have different prices for their service. Hence, budget should be spent in an efficient way while selecting the influencers. The retailer aims to maximize its profit while staying within the allowed budget.

Information cascades in social networks can be modelled by employing various diffusion models including Markov random fields [5], voter models [6], Independent Cascade Model (ICM), and Linear Threshold Model (LTM). Most common among the diffusion models in the literature are ICM and LTM.

LTM assumes that diffusion time steps are discrete. At any time, a node can be either active (i.e., influenced) or inactive. A node cannot become inactive later once it is active (i.e., a progressive model). Each node, in a way, contributes to activation of their neighbours. In LTM, each link is assigned a weight w_{vu} representing the influence of node v towards the target node u. Each node has an assigned threshold θ_u to get activated. The process starts with initially active nodes which serve as the seed nodes. At any time step t, for node u, if sum of influence weights on links originating from neighbouring active nodes exceed the randomly determined threshold θ_u, then u becomes active. The process runs until the time step where no more nodes get activated.

In ICM [7], on the other hand, node v activated at time t tries to activate its inactive neighbour node u only at time $t+1$. The attempt is successful with probability p_{vu}. Therefore, ICM is inherently a stochastic process.

Our Contribution

In this study, we make the following contributions:

- We define the new Targeted and Budgeted Influence Maximization in Social Networks under Deterministic Linear Threshold Model problem. This problem differs from the existing studies in the literature by (i) considering a deterministic diffusion model, (ii) extending the original Influence Maximization Problem [7] to a *targeted* version of the problem where nodes might carry heterogeneous profit values, and to a *budgeted* version of the problem where nodes might carry heterogeneous cost values for becoming seed nodes.

- We develop a new algorithm named Targeted and Budgeted Potential Greedy (TABU-PG) for the problem we defined. The algorithm employs a set of alternative methods for node selection and potential gain calculation. Some of the optional methods included in TABU-PG are taken from the literature to serve as benchmarks and the others are novel methods introduced in this work.

- We propose novel methods to enable TABU-PG heuristics to run on very large networks in a significantly shorter amount of time by trading between spread performance (i.e., total profit) and runtime.

- We propose new methods for generating influence weights for links; and threshold, profit, and cost values for nodes. In our opinion, in many cases, our methods reflect the real world dynamics more accurately than most widely employed methods in the literature.

- We provide empirical evaluations of TABU-PG heuristics and benchmarks such as closeness, betweenness, pagerank, strength, authority, hub, eigenvector, and random heuristics. With extensive computational experiments we show how all heuristics perform with 8 different datasets on 4 different real-life networks.

The paper is structured as follows. In Section 2, we review how Influence Maximization Problem emerged and developed in the literature, along with a comparison with our study. In Section 3, we provide a formal definition of the problem, present our TABU-PG algorithm, and describe the dataset generation methods we employ. In Section 4, experimental results and discussion are given. The conclusion and final remarks are given in Section 5.

2. Related Work

Domingos and Richardson [5] popularized the concept of *network value* of customers. By approaching the market as a set of connected entities rather than independent entities, they shifted the paradigm to considering the extra value which might emerge as a result of influences between entities instead of considering only the intrinsic value of each entity. Their study introduced the fundamental problem of *Influence Maximization*, that is how to choose seed nodes so that particular influence spread functions in social networks are maximized.

Kempe et al. [7] formulated the problem that is posed by [5] as a stochastic discrete optimization problem under Independent Cascade Model (ICM), Linear Threshold Model (LTM), and their special cases. As a solution, their work proposes a greedy approximation algorithm: the General Greedy Algorithm. Since they employ a stochastic diffusion model by randomizing the spread of influence instead of employing a deterministic diffusion model, they are able to obtain a submodular objective function. Using properties of submodular functions, they obtain an approximation guarantee of $(1 - 1/e)$ for their algorithm. Given a graph $G(V, E)$, size of the desired seed set (i.e., budget) k, and a submodular and monotone influence spread function f; at each iteration, they choose a seed v and add it to the seed set S such that $f(S \cup \{v\}) - f(S)$ is maximized. The algorithm halts when k nodes are selected as seeds.

The General Greedy Algorithm requires k iterations. At each iteration, the algorithm estimates the influence spread of $S \cup v$ for every $v \notin S$. Obtaining an accurate estimate of the influence spread requires a large number of Monte Carlo simulations, typically 10,000 times [8]. Therefore, the original greedy algorithm takes a very long time to complete. For instance, it takes multiple days to select 50 seeds in a network of 30K nodes [9]. To sum up, the original greedy algorithm is inefficient for two reasons: (i) Monte Carlo simulations are called kn times given a budget k and number of nodes in the network n, and (ii) large number of Monte-Carlo simulations are required. In order to improve the long running time of the original greedy algorithm, [7] is followed by a number of studies which suggested modifications on their original greedy algorithm or proposed different heuristics. [10, 11, 12] focus on improving upon the first limitation by employing lazy evaluation techniques. [13, 14, 9, 8] try to overcome the second limitation by developing heuristic algorithms which do not require Monte Carlo simulations.

Targeted Influence Maximization

Most of the initial studies in the literature estimate the spread of influence in terms of number of activated nodes. However, in practice, each node carries a different value for the marketer. The promoted product might be relevant for only nodes of certain types or nodes in certain locations. The expected profit might be different for each node since the nodes' perceived values of the product might differ or because of the varying purchasing power of the nodes. Thus, it is an obvious direction to assign heterogeneous profit values to nodes and modify the objective function to account for these values.

[15, 16, 17, 18, 19] study the Targeted Influence Maximization (TIM) problem under ICM or LTM. All of the algorithms modify the objective function to include profit or target fitness values. In assigning such values, they either benefit from the data already available in the datasets or generate values arbitrarily. To the best of our knowledge, there is no work which studies Targeted Influence Maximization Problem under a deterministic diffusion model.

Budgeted Influence Maximization

Many studies in the literature assume that initial activation costs of potential seed nodes are uniform, therefore the problem in such studies results in having a cardinality constraint in selection of seed sets. However, in practice, the nodes (e.g., accounts in social networking websites) have varying influence capabilities, self-perceived values, and different pricing strategies for becoming a seed node. Consequently, the nodes are expected to have heterogeneous initial activation costs. Hence, Budgeted Influence Maximization (BIM) problem focuses on the case where nodes have heterogeneous initial activation costs and the budget is monetary rather than an integer count.

[10, 20, 21, 22] study the problem of Budgeted Influence Maximization under ICM or LTM. Briefly, there are four general methods employed in seed selection: choose the node with (*i*) highest density, (*ii*) highest marginal gain, (*iii*) highest marginal gain above a certain density threshold, (*iv*) highest density above a certain marginal gain threshold. In a sense, the last two methods are a hybrid of the first two methods. To the best of our knowledge, there is no work which studies Budgeted Influence Maximization Problem under a deterministic diffusion model.

Deterministic Linear Threshold Model

Under ICM and LTM, the two most common diffusion models in the literature, it is NP-hard to determine the optimum for the Influence Maximization [7]. In both models, the exact computation of influence spread is shown to be #P-hard [14], but theoretical approximation guarantees can be obtained in both cases by employing the General Greedy Algorithm. This is a result of the influence spread being a monotone and submodular function. Both spread models are stochastic in nature. In ICM, the randomness is naturally obtained since activation of a node depends on influence probabilities on edges. In LTM, the randomness is obtained by assigning nodes random thresholds between 0 and 1 at each time step. Further, [7] showed that an equivalency can be established between ICM and LTM.

In practice, with availability and abundance of data, information on thresholds can be learned via surveys and data mining techniques making it a less randomized process. This leads to a Deterministic Linear Threshold Model (Deterministic LTM), where threshold values can be viewed as inputs to the model instead of assuming a random threshold function. For LTM, this approach is the deterministic version of the problem where the influence spread is based on deterministic rules and it makes great differences on the properties of the problem.

On the other hand, in contrast to models with randomized propagation, the number of activated nodes is not a submodular function of the target set when the threshold values are fixed [7]. Although the exact computation of the influence spread under Deterministic LTM can be solved in polynomial time [23], Lu et al. [24] showed that there is no $n^{1-\epsilon}$ factor polynomial time approximation for the Influence Maximization Problem under Deterministic LTM, unless $P = NP$.

Deterministic LTM is employed mainly in two problem types in the diffusion literature. [25, 26, 27, 28, 29] study the *set cover* problem where the aim is to find the minimum-sized set influencing all nodes in the network. [30, 31, 32] and we study the *maximum coverage* problem where the aim is to influence as many nodes as possible given an upper limit on size of the seed set.

[25, 26, 27, 28] study the Minimum Positive Influence Dominating Set problem, its complexities and employ greedy algorithms to find such a seed set. The diffusion model in those studies can be viewed as a Deterministic LTM with uniform weights and uniform thresholds. Askalidis et al. [29] seek to understand, given a subset of nodes called as a snapshot, whether there exists a seed set of size at most k which can result in the activation of the given snapshot under variations of Deterministic LTM.

Acemoglu et al. [30] study the dynamics of Deterministic LTM and explore whether different types of networks in terms of topological structures make diffusion more easy in them. Xu [31] proposes a sparse optimization technique with a linear algebraic approach for the seed selection problem. Swaminathan [32] proposes Threshold Difference Greedy (TDG) Algorithm for the Influence

Maximization Problem under Deterministic LTM. The main idea behind the algorithm is as follows. When comparing nodes in the seed selection steps, a node should also be rewarded for partially influencing other nodes. Such a reward is obtained by calculating marginal gain based not only on activated nodes but also the nodes that are partially influenced. Thresholds of the partially influenced nodes are decreased by the influence exerted on them. Consequently, it will be easier to activate them in the next iterations.

The deterministic activation of nodes was first thought to be a highly simplistic view in the Influence Maximization Problem [7]. However, in line with the ever-increasing speed of the technological advancements in terms of data mining techniques and data collection abilities; it becomes more feasible to predict people's behaviour in product adoption which in turn results in the estimation of node threshold values and link influence weights in LTM. Once these values are learned, better fitting algorithms can be enabled for the deterministic version of LTM; replacing most of the algorithms in the literature which assume stochastic diffusion models.

In this paper, we study the Targeted and Budgeted Influence Maximization Problem under the Deterministic LTM. Since the Influence Maximization Problem under Deterministic LTM considered in [23] is a special case of our problem where profit is equal to 1, and cost of activating a node is 1, for all nodes, their conclusions also apply to our problem and it cannot be approximated to within any non-trivial factor. Table 1 shows comparison of the problem definition in our work with that of existing studies in the literature in terms of diffusion model, diffusion type, extensions, and problem type. A mark symbolizes relatively strong focus on the given aspect. Our work is given in the last row and up to our knowledge it is shown to be novel in the literature according to the given classifications.

Table 1: Problem Definition Comparison with the Related Work

Ref.	LTM	Deter.	Targ.	Budg.	Inf.Max.
[5]	-	-	-	-	X
[7]	X	-	-	-	X
[9]	X	-	-	-	X
[10]	X	-	-	X	X
[11]	X	-	-	-	X
[12]	-	-	-	-	X
[13]	-	-	-	-	X
[14]	-	-	-	-	X
[8]	X	-	-	-	X
[20]	-	-	-	X	X
[21]	-	-	-	X	X
[22]	X	-	-	X	X
[15]	-	-	X	-	X
[16]	X	-	X	-	X
[17]	-	-	X	-	X
[18]	-	-	X	-	X
[19]	-	-	X	-	X
[30]	X	X	-	-	-
[26]	X	X	-	X	-
[27]	X	X	-	-	-
[28]	X	X	-	-	-
[29]	X	X	-	-	-
[31]	X	X	-	-	X
[32]	X	X	-	-	X
This study	**X**	**X**	**X**	**X**	**X**

3. Methodology

3.1. Formal Problem Definition

Let $G = (V, E)$ be a directed network where V is the set of nodes with $|V| = n$ nodes, and E is the set of links with $|E| = m$ links. Each node $v \in V$ is associated with a threshold value θ_v, an activation cost for being a seed node c_v, and a profit value p_v. Each directed link has an influence weight i_{uv} representing the amount of influence node u has on node v. The budget is denoted by B.

At any time step, a node can only be in one of the two states, inactive or

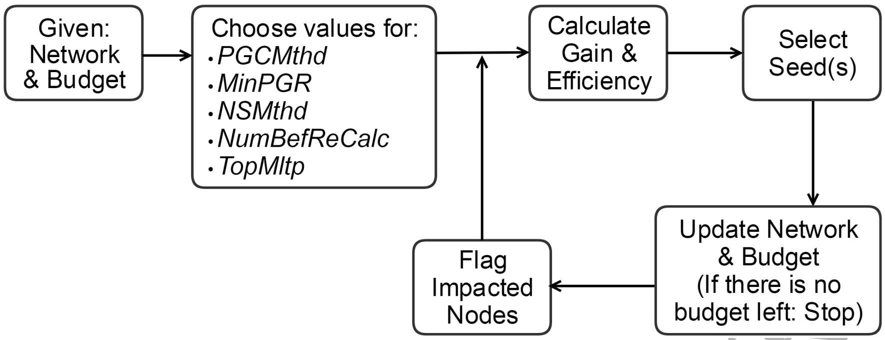

Figure 1: TABU-PG Flowchart

active, represented by $\sigma_v \in 0, 1$. $f(v)$ describes transition of the state of node v and it is solely based on the Deterministic LTM.

The problem aims to find a set of seed nodes S under the constraint $\sum_{v \in S} c_v \leq B$, such that activating the nodes in S is expected to maximize the total profit $P = \sum_{v \in V/S} p_v \sigma_v$ over the social network.

3.2. Targeted and Budgeted Potential Greedy (TABU-PG) Algorithm

We first provide a flowchart (in Figure 1) and a short outline of the algorithm below. It is then explained step-by-step in detail below along with an illustrative numeric example. TABU-PG works in an iterative and greedy fashion. The procedures in each step of the algorithm are managed by alternative methods provided in TABU-PG leading to different versions of the general algorithm. A list of all the methods employed are given in Table 4 in Section 4. In our Experimental Results section, we present a comparative analysis of these different heuristics.

Outline of the TABU-PG algorithm

Input: Influence weights, thresholds, costs, profits, budget, $MinPGR$, $PGCMthd$, $NSMthd$, $NumBefReCalc$, $TopMltp$

Output: Set of seed nodes, set of all influenced nodes.

Step 1 [GAIN CALCULATION]

Calculate actual and potential gain for each node.[1] $MinPGR$ determines a lower bound on the calculation of potential gain and $PGCMthd$ determines the weight of the potential gain.

Step 2 [SEED SELECTION]

Choose one seed with respect to one of the following methods, $NSMthd$: i) Maximum gain, ii) Maximum efficiency (gain/cost), iii) Hybrid of the first two methods.

Step 3 [NETWORK UPDATE]

Recalculate thresholds for each node, determine nodes whose gain values need to be re-calculated, and update the budget.

Step 4 If there is budget left, then go to step 1. Else, STOP.

GAIN CALCULATION

To identify the seed set, nodes are compared on measures that are based on nodes' *gain* values which is calculated as the sum of *actual gain* and *potential gain*. Actual gain of a node is calculated by measuring the increase in the total actual profit which emerges due to the node's capability of fully influencing (i.e., activating) neighboring nodes, and also other nodes by passing influence via the nodes it activated. For instance, assume that node u exerts influence towards node v whose current threshold is lower than the incoming influence from u. Therefore, u is able to activate v. If v is the only neighbor u can activate and v cannot activate any other node, then actual gain of u is equal to p_v. If v is also able to activate a single node w, then actual gain of u is equal to $p_v + p_w$, unless

[1] For future iterations; for all nodes *whose gain values need to be updated.*

w activates another node. Potential gain is related to the concept of partial influence [33], which is an important theme of our algorithm. Partial influence can be described as influence exerted by a node towards another node, which decreases the threshold of the latter but cannot eliminate it in full. In this case, the latter node is not immediately activated but is more likely to be activated in later iterations since it now has a lower threshold.

In TABU-PG algorithm, Minimum Potential Gain Ratio $MinPGR$ and Potential Gain Calculation Method $PGCMthd$ together specify how partial influences are accounted for while calculating the potential gain values for nodes.

Only the partial influences satisfying the constraint of i_{uv} to θ_v ratio being greater than or equal to $MinPGR$ are accounted for. If the ratio is lower than $MinPGR$, the partial influence is ignored in calculation of potential gain. If the ratio is greater than or equal to $MinPGR$, the potential gain can be calculated by multiplying p_v, i_{uv}/θ_v, and Potential Gain Multiplier ($PGMltp$). Potential Gain Calculation Method ($PGCMthd$) specifies how $PGMltp$ is calculated. In this work, we employ four methods for $PGCMthd$. In Method 1, $PGMltp$ is set to 0 thus effectively removing the potential gain from the algorithm. In Method 2, $PGMltp$ is set to 1.

In addition to Method 1 and 2, we propose two novel methods. In Method 3, $PGMltp$ is dynamically assigned the value of $1 - E/B$ each time, where E is the amount expensed so far. For instance, when half of the budget is exhausted, $PGMltp$ is set to 0.5. In Method 4, $PGMltp$ is dynamically set to $1 - (E/B)^2$ similar to the previous method. In this case, for instance, $PGMltp$ is set to 0.75 when half of the budget is exhausted. The intuition behind the third and fourth method is as follows. Since the potential gain represents the investment to the future profits, the value of $PGMltp$ should decrease when the chances of reaping these future profits decrease. As the number of future steps is limited by the remaining budget, the chances decrease and eventually converges to zero as budget is exhausted.

Therefore, letting g^a and g^p represent actual and potential gain, respectively, total gain is calculated as follows: $g = g^a + g^p(PGMltp)$.

SEED SELECTION

Node Selection Method ($NSMthd$) specifies the measure on which nodes will be compared to be selected as seeds, together with specifying any additional constraints. Since each node may have a different cost, choosing the node with the maximum gain is not the only reasonable option. We currently employ three node (seed) selection methods. Method 1 compares nodes solely based on gain and selects the node with maximum gain. Method 2 compares nodes solely on efficiency, that is the ratio of gain to cost g_v/c_v, and selects the most efficient node. This ratio of efficiency is also called density [21].

In addition, we also propose a novel third method. Method 3 is a combination of the first two methods. In this method, top three candidate nodes are selected based on efficiency (i.e., applying Method 2 but choosing three nodes instead of one). Then, among the three, the node with maximum gain (i.e., applying Method 1 but comparing only three nodes instead of many) is selected as the seed node. It is possible to combine the first two methods in other ways such as calculating scores for both methods and averaging them, or choosing top three nodes based on gain and selecting the seed among them based on efficiency. However, in our preliminary experiments they did not perform better than Method 2. Therefore, we do not include such additional methods in this work.

NETWORK UPDATE

Immediately after a node is selected as seed, the network and the budget is updated accordingly. The influence spreading from the seed node is propagated through the network; thresholds of the impacted nodes are reduced, and suitable nodes are activated. Therefore, the next seed node will not be selected among already active nodes and the gain calculations will be done with up-to-date threshold values and status of nodes.

Flagging Nodes for Gain Recalculation: Updates in the thresholds and status of nodes cause changes in the gain values of nodes which exert influence upon them. Hence, gain need to be recalculated for those nodes. For this purpose, during the network update step, nodes whose thresholds or status are updated are *marked*. Nodes which fully or partially, directly or via its cascades, influence the *marked* nodes are to be *flagged* for gain recalculation. To achieve this, first, nodes which have outgoing influence links to the *marked* nodes are flagged. Then, nodes which are capable of activating any *flagged* node are also flagged. The last step iterates until no new node gets flagged. This way, we are able to find all nodes which have derived gain from the nodes whose thresholds or status have changed.

Method 3 and 4 for $PGCMthd$ in GAIN CALCULATION requires potential gains to be modified at each iteration based on the remaining budget. By storing the potential gain before multiplication by $PGMltp$, we are able to update potential gain of a node without recalculating its actual gain and potential gain value in the gain calculation step.

In this way, in the next gain calculation step, gain values will be calculated only for nodes which are in the intersection set of flagged nodes and non-activated nodes, which together constitute the candidate pool.

An Illustrative Example for TABU-PG

To illustrate how TABU-PG works, consider the network in Figure 2. In this network, a directed arc from node i to j represents that node i can influence node j. Cost and profit values of each node are given in Table 2. $MinPGR$ is selected as 0. $PGCMthd$ is selected as the third method, that is decreasing the share of potential gain as budget is exhausted. Node threshold values are given inside the nodes. Influence weights are given on links. Nodes selected as seed are shown with a bold font type. For ease of display, we remove the thresholds from the figure once the node becomes active. The campaign budget is set to 5.

Table 2: Profit and cost values for each node.

	A	B	C	D	E	F	G	H
Cost	2	4	2	1	2	2	2	3
Profit	1	3	4	2	10	5	6	0

<u>*Iteration 1:*</u>

Gain values are calculated for all nodes. Potential gain and actual gain calcu-

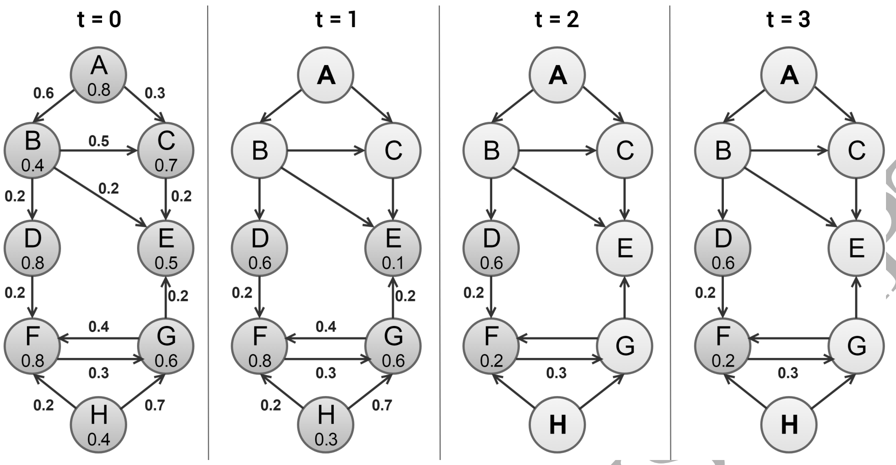

Figure 2: TABU-PG Illustration

lations are shown below. Total gain is calculated as ($g = g^a + g^p(PGMltp)$), where $PGMltp$ is 1 in the first iteration since no budget is spent yet, i.e., $E = 0$.

${g^a}_A = 3 + 4 = 7, \quad {g^p}_A = (0.2/0.8)2 + (0.2/0.5)10 + (0.2/0.5)10 = 8.5.$

${g^a}_B = 0, \quad {g^p}_B = (0.5/0.7)4 + (0.2/0.8)2 + (0.2/0.5)10 = 7.36.$

${g^a}_C = 0, \quad {g^p}_C = (0.2/0.5)10 = 4.$

${g^a}_D = 0, \quad {g^p}_D = (0.2/0.8)2 = 0.5.$

${g^a}_E = 0, \quad {g^p}_E = 0.$

${g^a}_F = 0, \quad {g^p}_F = (0.3/0.6)6 = 3.$

${g^a}_G = 0, \quad {g^p}_G = (0.2/0.5)10 + (0.4/0.8)5 = 6.5.$

${g^a}_H = 6, \quad {g^p}_H = (0.2/0.8)5 + (0.2/0.5)10 + (0.4/0.8)5 = 7.75.$

When selecting the seed node, we have three options as explained in Section 3.2. For this illustration, we will employ $NSMthd = 2$, the second method that is choosing the most efficient node. Accordingly, the efficiency values are found as $A : 15.5/2 = 7.75$, $B : 7.36/4 = 1.84$, $C : 4/2 = 2$, $D : 0.5/1 = 0.5$, $E : 0$, $F : 3/2 = 1.5$, $G : 6.5/2 = 3.25$, and $H : 13.75/3 = 4.58$, by dividing total gain by cost of the node. A is selected as the first seed node since it has the largest efficiency.

As the thresholds for neighbors of A are lowered, first B and then C becomes active. The budget becomes 3. Next, potential and actual gain values will be calculated only for those nodes which are not already activated and whose gain value calculations are affected by the updates in the network. Therefore, in the second iteration, potential gain and actual gain values will be calculated only for G and H since G (H) took into account influencing E (G) in its gain calculation in the previous iteration.

<u>*Iteration 2:*</u>

$g^a{}_G = 10, \quad g^p{}_G = (0.4/0.8)5 = 2.5.$

$g^a{}_H = 6 + 10 = 16, \quad g^p{}_H = (0.2/0.8)5 + (0.4/0.8)5 = 3.75.$

Total gain values ($g = g^a + g^p(PGMltp)$) for all nodes are calculated this time by multiplying potential gains by $1-(2/5) = 0.6$ since we employ $PGCMthd = 3$. Actual and potential gains for D, E and F are the same as the first iteration, however, total gains are re-calculated with the updated $PGMltp$. Then, efficiency values are found as $D : 0.3$, $E : 0$, $F : 0.9$, $G : 5.75$, and $H : 6.08$, by dividing total gain by the cost of the node. H is selected as the second seed node since it has the largest efficiency. The network is updated accordingly and there are no new active nodes. Since there is no budget left, the algorithm stops.

The set of seed nodes are $\{A, H\}$ and the final set of active nodes are $\{A, B, C, E, G, H\}$ with a total profit of 24.

Scaling for Larger Networks

In order to make the algorithm scalable to very large networks, we limit the number of calculations in the two major steps of the algorithm. $NumBefReCalc$ specifying the number of nodes to select before recalculating gain values, and $TopMltp$ specifying the multiplier for determining the number of top nodes are defined. $NumBefReCalc$ specifies how many nodes are to be selected as seed nodes based on the current gain values for nodes; before recalculating the gains for the updated network. When $NumBefReCalc$ is set to 1 which is the default, gains are calculated before each time a new seed node is selected. When it is set to Inf, gains are calculated only once at the beginning, and all seed nodes are selected in the first iteration based on these gain values. When, for example, it is set to 5; after a gain calculation, up to five more nodes can be selected before recalculating the gain values.

$TopMltp$ takes part in determining the number of top nodes for which gains will be calculated for after the first gain calculation (which is done for all nodes). The intuition is that if a node's gain or efficiency is very small at first; it is very unlikely that the gain or efficiency values which will be calculated for that node will be large enough for that node to be selected as a seed at later steps. Therefore, the idea is to calculate gain values only for the nodes which are more likely to be selected as seed nodes.

Before recalculating the gain values, number of top nodes which gain will be calculated for is calculated based on $TopMltp$, $NumBefReCalc$, and current size of S which is the current seed set at that time. The number of top nodes is equal to $(|S| + NumBefReCalc)TopMltp$. $NumBefReCalc$ is included in the formula to ensure that there will be enough number of nodes to consider when $NumBefReCalc$ is larger and thus gain is calculated less frequently.

If Method 1 is employed for $NSMthd$, all nodes are sorted based on their initial gains and the calculated number of top nodes are selected. If Method 2 or Method 3 is employed for $NSMthd$, nodes are sorted based on their efficiency instead of gain. Utilizing this method, we limit the candidate pool for seed nodes to a number of top nodes instead of all nodes.

Introducing these methods enables a trade-off between runtime and final influence spread, therefore making it possible to run the algorithm on very large graphs in a reasonable amount of time. Note that, theoretically, there could be large improvements in runtime without any worse performance in influence spread results at all.

To give a numerical example of size reduction using these two parameters, consider the following case. Assume that $NumBefReCalc$ is set to 5 and $TopMltp$ is set to 10. In the first iteration, gain will be calculated for all nodes. The nodes will be sorted based on their gain (or efficiency) values and stored in a list L. Among all nodes, 5 nodes will be selected as seed nodes. In the second iteration, the candidate pool will be limited to the top $(5 + 5)10 = 100$ nodes in L. The gain values will be calculated only for the candidate pool. From the candidate pool, 5 nodes will be selected as seed nodes. Similarly, in the third iteration, the candidate pool will be limited to the top $(10 + 5)10 = 150$ nodes in L, rather than considering all inactive nodes in the network.

3.3. Dataset Generation

For the empirical analysis, the following graphs are employed: Epinions [34, 35], Academia [36, 37], and Pokec [34, 38]. In addition, we crawled the underlying social network of Inploid[2]. All graphs are directed.

Epinions is a consumer review website where users can share their reviews for a variety of products. In order to prevent deceiving reviews, a trust system is put into place where users can specify whether other users are trustworthy or not. It results in a social network where nodes are the users and directed links are indicators of trust between the users. Academia is a social networking website for academics where academics can share papers and follow each other. Pokec is a Slovakian online social networking website where users can share information about themselves, post pictures on their profiles, and chat with other users. The information about users include age and gender. The dataset covers the whole network. Inploid is a social question and answer website in Turkish. Users can follow others and see their questions and answers on the main page. Each user is associated with a reputability score which is affected by feedback of others about questions and answers of the user. Each user can also specify their interests in topics. We crawled this network in June 2017. In addition, for each user, reputability scores and top five topics are included in the dataset. The dataset contains all users as of the crawling date.

The following data preprocessing procedures are applied to all graphs. Any self-loops and multiple (i.e., recurring) links are removed from the graph. Largest connected component in each network is found and other components are removed from the network. This is preferred because disconnected components

[2]The anonymized dataset can be downloaded from https://furkangursoy.github.io/datasets/

are, in a sense, like multiple different networks rather than a single network. Additional data preprocessing steps are taken for Pokec graph. The nodes whose follower information was not public, approximately one third of all nodes, are excluded from the network. The information on resulting graphs before and after data preprocessing operations are performed are summarized in Table 3.

Table 3: Number of Nodes and Links for each Graph

	Before		**After**	
Graphs	**# of nodes**	**# of links**	**# of nodes**	**# of links**
Epinions	75,879	508,837	75,877	508,836
Academia	200,169	1,398,063	200,167	1,397,620
Inploid	39,750	57,276	14,360	57,100
Pokec	1,632,803	30,622,564	1,080,251	14,662,846

Generation of Influence Weights

Influence weights on links represent the amount of influence one node has on another. For some networks, these values might be available in the dataset in some forms due to the nature of the given social network (e.g., trust degrees). However, in most cases, social networks do not have mechanisms to assign such values to the links. Therefore, it is necessary to assign proper values as influence weights to the links.

Although influence models and influence weight determination play a significant role in influence maximization problems, they attracted less attention from researchers compared to algorithm design for the problem [39]. In the literature, influence weights (or influence probabilities in the case of ICM) are assigned in the following ways: fixed value, arbitrary, or ratio model. In fixed value method, all links are assumed to have the same fixed weight such as 0.05 or 0.1. When this method is employed [15], it effectively ignores any differentiation in influence capabilities of links; which is not a well representation of mechanisms in the real world. In arbitrary selection method [16, 20], influence weights are sampled randomly from a set of values such as $\{0.01, 0.05, 0.1\}$ or from an interval such as $[0, 1]$. This is a better representation of the real world compared to fixed value method since it acknowledges that different links might have different influence capabilities. In ratio model [7, 9, 32], influence weight is assigned by dividing 1 by the number of incoming links to target node. The equation for influence weight is given in the following formula: $i_{uv} = 1 / v_{indeg}$. This is the most widely used method in the literature.

When the fixed value or arbitrary selection method is employed; a node becomes active based on count of its neighbors which are already activated. For instance, at least five neighbors with influence weight of 0.1 on the links are required to activate a node with a threshold of 0.5. On the other hand, in the ratio model, a node becomes active based on proportion of its active neighbors instead of count. For example, to activate a node with a threshold of 0.5, at least 50% of its neighbors are required to be active given that links have equal weights.

The methods which are based on count makes it relatively difficult to activate nodes with smaller degrees. On the contrary, the methods which are based on proportion makes it difficult to activate nodes with larger degrees (i.e., nodes with many incoming influences). Considering this trade-off, we develop a novel hybrid method which is a fusion of arbitrary selection method and ratio model.

In our proposed method; first, average degree of the network is calculated by dividing number of links to number of nodes. Then, 1 is divided by the average degree and a fixed value is found. For each link, this fixed value is multiplied by a value sampled from $\{a_1, a_2, ...\}$ with respective probabilities of $\{p_1, p_2, ...\}$. In effect, it is similar to arbitrary selection method. However, we introduce sampling probabilities and the practice of dividing 1 by the average degree of the network. Then, for each link, geometric mean of the result of the above method and result of the ratio model is calculated and assigned as the influence weight. For example, in the case where the influence weight resulted from ratio model is 0.2, and the influence weight resulted from the above method is 0.45; our hybrid model results in influence weight of 0.3, their geometric mean.

In experimental studies, we use both ratio model (odd numbered experiments) and our proposed hybrid model (even numbered experiments).

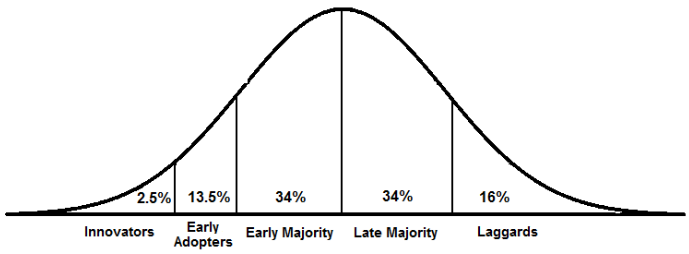

Figure 3: Five Groups of Consumers in Adoption of Innovation

Generation of Threshold Values

Node thresholds are not readily available neither in the datasets we use nor in any dataset in the literature to the best of our knowledge. Therefore, node thresholds need to be synthetically generated for experimental purposes.

In most of the literature, thresholds are assigned randomly between 0 and 1 to satisfy the submodularity requirement for the original LTM. In studies about PIDS problems, thresholds are assigned a fixed value, usually 0.5. In [32], fixed values 0.8 and 0.5; or random values between 0.1 and 0.9, or between 0.3 and 0.7 are used. Most of the studies use the ratio model in assignment of influence weights; and in such models, a node with a threshold above 1 can never be activated.

Since submodularity does not hold in Deterministic LTM, we are not limited to drawing thresholds randomly between 0 and 1. On the other hand, assigning all nodes a same fixed value is an oversimplification. Instead, we develop a new approach which mimics the real world dynamics of diffusion of innovations.

As Rogers put it in his seminal work [3], there are five groups of consumers when it comes to adoption of innovation: innovators (2.5%), early adopters (13.5%), early majority (34%), late majority (34%), and laggards (16%). Although market shares are given for five discrete groups, they do not follow a discrete distribution but a normal distribution, as illustrated in Figure 3. The continuous distribution provides that, for example, a person in the early majority group can be closer to early adopters or late majority depending on its threshold. In all the experiments, we employ a normal distribution with a limit on lowest value (i.e., truncated normal distribution [40]) with the aim of mimicking the real world dynamics.

It is worth noting that values of influence weights or thresholds become meaningful only in comparison to values of the other. For instance, a threshold of 0.8 can be seen as relatively high when most of the influence weights are smaller than 0.1, however it can be seen as relatively low when most of the influence weights are greater than 0.4. Thus, influence weights and thresholds should be viewed as strongly interrelated components.

Generation of Cost Values

In the literature, studies on BIM problem embraced the following ways of assigning cost values to nodes. [21] employs two methods. First one is to assign a fixed uniform cost to all nodes. The second method is assigning costs based on node degrees. Similarly, [10] use fixed uniform costs to all nodes (unit cost model) as the first model, and assign costs to blogs (i.e., nodes) based on the number of blog posts they contain as a second method. [20] employ the method of selecting costs randomly from an interval in addition to employing unit cost model.

Instead of assigning random or fixed values as costs for nodes, we develop a new method similar to that of [21], that considers the indegree (i.e., number of outgoing influences) of a node while estimating the cost of that node.[3] The intuition is as follows. In typical social networks, nodes (e.g., users) are not likely to know their true network value. Instead, we assume, a node's self-perceived value is mostly based on the number of its followers. It is a simple metric on which users can compare themselves with others, and estimate their own value in the market.

However, degree is not a deterministic factor by itself. In order to normalize the cost values, square root of indegree is used instead of indegree. The found

[3] A node with a large indegree value means that number of its followers is large. Direction of an edge and direction of the influence on that edge is opposite and should not be confused.

value then is multiplied with a random value between a and b, to represent the variation in users' methods of self-perception. We specify a fixed value z added to costs of all nodes, which represents any possible fixed costs in real life (e.g., cost of time required for communication with any node, legal costs, cost of sample products, and etc.). The formula for cost calculation is as follows $c_v = \sqrt{v_{indeg}} rand(a, b) + z$.

In all experiments except for experiments on Inploid, cost values are assigned according to following formula we suggest: $c_v = 1 + \sqrt{v_{indeg}} rand(min = 0.5,\ max = 1.5)$. For experiments on Inploid network, cost values are assigned according to following formula: $c_v = \sqrt[3]{v_{indeg}} + \sqrt[3]{v_{rep}} + 1$. Differently from the formula used in other datasets, this includes v_{rep} which represents reputability scores of users which is readily available in the dataset. Cube root is employed for normalization purposes. Overall, this cost assignment method might reflect a case where users determine their costs by their reputability scores along with the number of their followers.

Generation of Profit Values

Profit values can be used for targeting certain demographics where they are generated according to fitness of nodes to target demographics. They can also be used simply for profit maximization by generating profit values based on estimated profit (e.g., socioeconomic status of nodes) from the nodes. Another option is to combine these two approaches to represent both fitness to target demographics and estimated profit values.

In the literature, profit values are assigned to nodes in the following ways. [16] employs a product-user network and derives profit values by combining the item ratings of users and prices of the items. [19] uses a location based social network and derives profit values (or so called target fitness values) by measuring the distance of users to the target location. [15] employs a movie-actor network and creates profit values based on the genres of the movies in which actors have played. [17] studies online advertising and assigns profit values based on similarity between the two term vectors they created for advertisements and users. [18] uses profile data of users to target users in certain categories. Generating profit values from the network is not always possible when the dataset lacks relevant information. In those cases, for experimental purposes, profit values need to be synthetically created.

In this paper, we consider the following generation methods. Profits can be drawn from a discrete distribution $\{a_1, a_2, ...\}$ and $\{p_1, p_2, ...\}$ where p_1 is probability of selecting a_1 and so on. For instance, if a campaign only targets the males living in urban areas, then a_1, a_2 and p_1, p_2 should be 1, 0 and 0.25, 0.75 respectively, given that half of people live in urban areas and male to female ratio in urban areas is 1. Profit can also be generated based on the assumed continuous profit distribution. Since the distribution is highly dependent on the product and context; for simplicity, we will assume a log-normal distribution as a representation of income inequality. A third method which combines the discrete distribution method and continuous distribution method can be achieved by multiplying the outputs of the two.

In Epinions and Academia networks, unlike Inploid and Pokec, there is no demographic or other types of information present for generating profit values. Thus, in experiments with Epinions and Academia, the values are assigned by multiplying the following two values: a selection in $\{0, 1, 2, 3\}$ with respective probabilities of $\{0.25,\ 0.25,\ 0.25,\ 0.25\}$, and a random selection from the log-normal distribution of $ln(mean = 1,\ sd = 0.3)$. The first component aims to represent different demographics (e.g., four equal sized target demographics). It is assumed that one group does not bring any profit at all, and the other three groups have respective importance degrees (e.g., profit potentials) of 1, 2, and 3. The second component is assumed to represent the income distribution among people in the network, creating further variations in profit values.

For experiments on Inploid network, profit values are assigned in the following way. There exist over 3000 unique topics which users have shown interest in. STEM (Science Technology Engineering Mathematics) related topics are manually flagged by us. In the dataset, a user can have at most 5 associated topics. Number of STEM-flagged topics are counted and assigned as profit values. For example, if a user is interested in 3 STEM-related topics, its profit value is assigned as 3. For users who did not specify any topic on their profile or who are not interested in any STEM-related topic are assigned a profit value of 0.5. Overall, this profit assignment method might reflect a case where a product's primary target group is people who are interested in STEM.

In Pokec social network, not all users have specified their ages on their profile pages. To fill the missing age data, random integer values between 18 to 45 are assigned. Profit values are assigned by targeting specific demographics with specified importance values. Females aged between 18 to 34 are assigned the profit value of 5, males aged between 18 to 24 are assigned the profit value of 3, males aged between 25 to 34 are assigned the profit value of 2, males aged between 35 to 44 are assigned the profit value of 1, and the rest is assigned the profit value of 0. The values are created based on an arbitrary hypothetical case where given demographics carry specified degrees of importance or potential profits for the marketer.

Determining endogenous parameters for the influence weights, thresholds, cost and profit values is a challenge. In addition, there is the complexity of data availability along with the complete network structure information. However, since Influence Maximization problems are solved for given parameters, we view parameter generation to occur in practice as a step before running our proposed algorithm. Overall, TABU-PG algorithm is agnostic to parameter values and designed to work with any set of parameter values, generalizing our solution methodology to a variety of settings.

4. Experimental Results

We present the performance of our algorithm with experimental results. An experiment is performed for each generated dataset.[4] Experiment 1 and 2 are for Epinions, Experiment 3 and 4 are for Academia, Experiment 5 and 6 are for Inploid, and Experiment 7 and 8 are for Pokec networks.

[4]All experiments are run on a computer with Intel Core i5-5200U CPU @ 2.20 Ghz, and 8 GB memory.

For each experiment, strength[5], closeness, betweenness, pagerank [41], hub [42], authority [42], and eigenvector [43] heuristics are employed as benchmarks in addition to the random heuristic where seed sets are selected randomly. When calculating the centrality scores to serve as benchmarks, directions and weights of the edges are considered. For each node, the obtained centrality scores are then divided by the cost of that node to obtain an efficiency score. In all benchmarks, obviously excluding the random heuristic, nodes are selected as seeds based on the obtained efficiency scores. Further improving the benchmark heuristics, seed nodes which would still be activated by the diffusion process at some time step (even if they were not seeds) are removed from the seed list. In this way, budget is used more efficiently.

Initial experiments showed that closeness and hub heuristics perform significantly poor. We were able to obtain better performance for closeness and hub heuristics once link weights and node costs are not taken into account. Hence, we assumed unit-weights and unit-costs while applying closeness and hub heuristics. In Experiment 1, we assumed unit-weight for calculation of eigenvector centrality since the calculations did not converge in 100,000 iterations. For experiments on Pokec, betweenness and closeness scores are considered without considering the weights since it takes multiple days to estimate those centrality scores when weights are considered. Cutoff points[6] are set as 3 for all except for the experiments on Academia and Pokec where it is set as 2. In calculation of PageRank score, for all experiments, damping factor[7] is set as 0.85 which is an assumed default value in most PageRank applications.

$MinPGR$, $PGCMthd$, $NSMthd$, $NumBefReCalc$, and $TopMltp$ manage which methods are to be employed in the algorithm. Therefore, different method selections result in different TABU-PG heuristics sharing the same framework.

[5] Since networks are directed, in-strength is calculated. Higher indegree equals to higher number of followers. In-strength is the sum of inward link weights.

[6] A cutoff point is the maximum path length to consider when calculating the betweenness or closeness score.

[7] Please refer to [41] for details.

Table 4: Methods Used in the Experiments

For Step 1	$MinPGR$: **Minimum Potential Gain Ratio**	
	x	if i_{uv} to θ_v ratio lower than x, the potential gain is ignored.
	$PGCMthd$: **Potential Gain Calculation Method**	
	1	$PGMltp$ is set to 0, effectively removes the potential gain
	2	$PGMltp$ is set to 1
	3	$PGMltp$ is set to $1 - (E/B)$
	4	$PGMltp$ is set to $1 - (E/B)^2$
For Step 2	$NSMthd$: **Node Selection Method**	
	1	selects the node with maximum gain
	2	selects the node with maximum efficiency
	3	a hybrid of Method 1 and Method 2
For Scaling	$NumBefReCalc$: **Number of nodes to select before recalculating**	
	Inf	all seed nodes are selected without recalculating gain values
	x	up to x nodes are selected before recalculating gain values
	$TopMltp$: **Multiplier for determining number of top nodes**	
	Inf	candidate pool is not limited by top nodes
	x	candidate pool is limited, determined by multiplier x

The methods are briefly explained in Table 4.

When displaying different settings, values are added as a suffix to the name of our algorithm. For example, TABU-PG_2_1_0.05_5_Inf states that our algorithm utilizes Method 2 in node selection (i.e., $NSMthd$), utilizes Method 1 in potential gain calculation (i.e., $PGCMthd$), employs minimum potential gain ratio of 0.05 (i.e., $MinPGR$), chooses 5 seed nodes before recalculating gain values (i.e., $NumBefReCalc$), and does not put a limit on maximum number of nodes to calculate gain for (i.e., $TopMltp$). Whenever the last two are not displayed when presenting a TABU-PG heuristic, the default values of 1 and Inf should be assumed.

There are only 3 heuristics employing Potential Gain Calculation Method 1 (i.e., ignoring potential gains altogether) since Minimum Potential Gain Ratio does not play any role when potential gains are ignored altogether, therefore such heuristics are not multiplied for different values of $MinPGR$. There are 12 heuristics for each Potential Gain Calculation Method 2, 3, and 4. Thus, a total of 39 heuristics are initially presented for each experiment except for

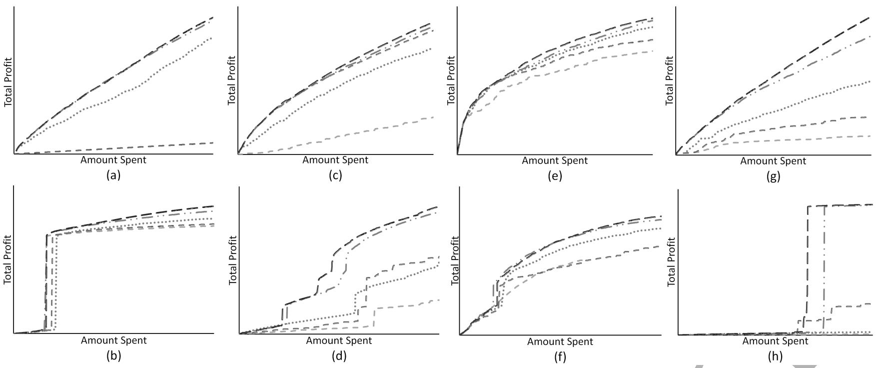

Figure 4: Detailed diffusion results for selected heuristics in Experiments 1-8 ((a)-(h), respectively).

experiments on Pokec where a smaller number of heuristics are presented due to the fact that experiments on Pokec take long time to complete. For example, a single TABU-PG heuristic takes more than 3 hours in Experiment 7, and more than 11 hours in Experiment 8. After a set of initial experiments, to obtain a moderate size spread which is helpful in better presenting computational results, we set the budget as 3000 for all experiments.

General Results

For each experiment; the best, median, and worst performing TABU-PG heuristics; and best and median performing benchmark heuristics are selected. Detailed diffusion results of the selected five heuristics in each experiment are given in Figure 4. x-axis shows the amount spent (i.e., E), and y-axis shows the total profit obtained. Scale of x-axis is from 0 to 3000 in all charts. On the other hand, scale of y-axis differ for each chart since the influence spread varies significantly in each experiment.

As it can be seen in Figure 4, total profit dramatically increases after reaching a certain point in even numbered experiments. It is a reflection of the *tipping point* phenomenon. The hybrid influence weight generation method creates tipping points in the network whereas the ratio model which is the most widely used method in the literature rather results in a decreasingly growing or a linear influence spread. This comparison further supports our claim that the hybrid model which is introduced in this work is a better representation of the real world because tipping points exist in most real life networks.

Some of the algorithms in the figure reach tipping points before others in terms of exhausted budget. Therefore, in a case where the budget is close to the tipping points, tuning between alternative methods can result in extraordinarily large improvements.

Table 5 summarizes performances of all heuristics with respect to final total profit reached by influence spread. Experiments are named with letter E suffixed by the experiment number. For each experiment, heuristic performances are presented relative to the worst performing TABU-PG heuristic, which is denoted by 100%, in the given experiment. For each experiment, results of best performing and worst performing TABU-PG heuristics are shown in bold font. Average performances (μ) and standard deviation of performances (σ) are calculated without considering experiments on Pokec since not all heuristics are run on Pokec. The heuristics are sorted based on their average performances. Average heuristic rankings are given without considering experiments on Pokec.

When benchmarks are compared among themselves, strength and pagerank heuristics are the best two performing heuristics on average. However, benchmarks do not guarantee a consistent performance in different experiments as evidenced by the higher standard deviation the benchmarks have in comparison to that of TABU-PG heuristics, especially when standard deviations are viewed relative to respective average performances.

Node Selection Method 3 is utilized by four of the top five performing heuristics, followed by Method 2. When TABU-PG_3_1_0 and TABU-PG_3_2_0 are excluded, a heuristic which employs Method 1 never performs better than any heuristic which employs Method 2 or Method 3 in almost all experiments.

Potential Gain Calculation Method 4 is employed by top performing heuristics followed by Method 3. Top 15 heuristics employ either Method 3 or Method 4.

A significant and consistent change in performances of heuristics is not ob-

Table 5: Comparison of All Heuristics (%)

	E1	E2	E3	E4	E5	E6	E7	E8*	σ	μ	Avg. Rank
TABU-PG_3_4_0.1	116.2	110.4	123.1	179.8	106.6	110.8	-	-	25.3	124.5	4.67
TABU-PG_3_4_0	116.6	110.5	122.8	176.8	**106.8**	**111.4**	182.0	4073.4	24.1	124.1	5.00
TABU-PG_2_4_0	116.9	110.5	122.8	176.7	106.2	110.7	181.7	4078.5	24.2	124.0	6.17
TABU-PG_3_4_0.05	116.8	110.5	122.8	174.7	106.8	109.9	-	-	23.4	123.6	7.67
TABU-PG_3_4_0.2	115.5	109.8	123.2	174.9	106.2	110.8	-	-	23.6	123.4	8.33
TABU-PG_3_3_0	115.9	110.5	122.9	176.6	105.8	108.7	185.1	293.3	24.4	123.4	9.50
TABU-PG_3_3_0.05	115.3	110.3	123.0	175.6	105.7	108.7	-	-	24.1	123.1	11.33
TABU-PG_2_4_0.05	**117.0**	110.2	122.8	171.0	105.9	111.2	-	-	22.1	123.0	9.33
TABU-PG_2_3_0	114.9	110.0	122.9	175.9	105.7	108.7	**185.2**	**4080.4**	24.3	123.0	13.17
TABU-PG_3_3_0.2	115.4	109.6	**123.3**	174.4	106.1	107.3	-	-	23.8	122.7	12.33
TABU-PG_3_3_0.1	114.7	**110.6**	123.0	173.1	105.4	109.3	-	-	23.2	122.7	12.50
TABU-PG_2_3_0.1	115.1	110.2	123.0	175.0	104.8	108.1	-	-	24.1	122.7	15.17
TABU-PG_2_4_0.1	115.9	110.3	123.1	170.7	105.6	110.1	-	-	22.2	122.6	11.00
TABU-PG_2_4_0.2	115.8	110.1	123.2	169.8	106.0	110.3	-	-	21.8	122.5	10.50
TABU-PG_2_3_0.05	114.4	110.0	123.0	174.5	105.0	108.2	-	-	23.9	122.5	16.50
TABU-PG_3_2_0.2	115.7	106.4	119.3	178.9	105.7	109.1	-	-	25.7	122.5	12.67
TABU-PG_3_2_0	114.2	106.4	118.0	**180.1**	105.4	109.4	158.9	4063.4	26.2	122.3	15.50
TABU-PG_2_3_0.2	114.5	110.1	123.3	170.4	105.1	108.6	-	-	22.4	122.0	15.17
TABU-PG_3_2_0.1	116.7	106.3	118.5	175.4	105.8	109.1	-	-	24.4	122.0	13.00
TABU-PG_3_2_0.05	114.3	106.3	118.0	175.9	105.7	109.0	-	-	24.7	121.5	17.67
TABU-PG_2_2_0.2	114.3	106.6	119.3	174.1	104.4	109.3	-	-	24.1	121.3	18.67
TABU-PG_2_2_0.1	114.6	106.2	118.6	172.6	105.1	108.0	-	-	23.6	120.8	20.50
TABU-PG_2_2_0	111.6	106.4	118.0	174.2	104.9	108.1	159.5	4067.7	24.4	120.5	21.50
TABU-PG_2_2_0.05	114.8	106.3	118.1	169.2	104.9	108.6	-	-	22.3	120.3	21.00
TABU-PG_1_4_0.1	104.5	103.8	110.6	168.0	104.9	104.5	-	-	23.3	116.0	28.50
TABU-PG_1_4_0	105.0	104.1	109.7	167.4	103.5	104.2	118.6	4057.3	23.2	115.6	30.83
TABU-PG_1_3_0.1	104.2	104.9	111.3	164.3	104.4	103.7	-	-	22.0	115.5	29.00
TABU-PG_1_3_0.2	103.6	104.8	111.9	165.3	104.8	102.3	-	-	22.5	115.4	29.67
TABU-PG_1_3_0.05	105.9	105.6	111.3	162.7	103.6	103.3	-	-	21.3	115.4	29.33
TABU-PG_1_3_0	105.4	104.8	111.3	162.7	103.8	103.3	121.6	4057.8	21.4	115.2	30.33
TABU-PG_1_4_0.05	105.6	104.6	109.7	161.1	103.9	104.2	-	-	20.8	114.8	30.50
TABU-PG_1_1_0	105.5	106.0	112.0	159.6	104.0	101.4	126.5	**100.0**	20.3	114.8	30.33
TABU-PG_1_4_0.2	102.2	104.2	111.3	160.9	105.0	102.3	-	-	21.0	114.3	31.17
TABU-PG_3_1_0	111.2	100.5	123.2	135.4	103.9	100.8	184.9	121.7	12.9	112.5	28.33
TABU-PG_1_2_0.2	103.0	100.7	105.3	161.5	101.3	101.1	-	-	22.1	112.2	35.33
TABU-PG_1_2_0.05	101.1	100.4	101.9	163.4	100.0	102.4	-	-	23.2	111.5	35.00
TABU-PG_1_2_0	**100.0**	**100.0**	**100.0**	163.2	**100.0**	102.4	**100.0**	4061.3	23.4	110.9	36.33
TABU-PG_1_2_0.1	100.5	100.1	102.9	159.0	100.7	101.5	-	-	21.6	110.8	37.00
TABU-PG_2_1_0	111.0	101.8	123.2	**100.0**	103.3	**100.0**	184.8	122.0	8.3	106.5	29.50
Strength	104.0	95.0	115.5	28.5	90.1	83.2	94.2	49.7	27.7	86.1	40.50
PageRank	101.8	93.7	102.8	25.3	88.2	72.1	65.0	35.6	26.8	80.6	41.67
Closeness	79.7	93.6	21.9	108.3	81.1	69.6	22.0	31.0	27.0	75.7	42.50
Eigenvector	62.6	91.1	34.6	89.6	61.3	70.5	4.1	15.6	19.1	68.3	43.83
Betweenness	48.6	88.5	44.1	47.6	77.2	59.6	27.7	974.7	16.5	60.9	43.33
Authority	61.4	93.4	31.1	9.4	84.1	77.1	12.3	11.3	30.0	59.4	43.83
Hub	32.4	88.5	18.3	85.1	70.2	47.7	6.4	11.8	26.3	57.0	45.33
Random	8.5	0.5	2.4	1.6	4.7	5.8	3.0	5.9	2.7	3.9	47.00

*For Experiment 8, results are reported for heuristics with $TopMltp$ 10 instead of Inf.

served when $MinPGR$ is set to values other than the default value of 0. However, it results in a reduction in performance for heuristics which employ a combination of Node Selection Method 2 and 3, and Potential Gain Calculation Method 3 and 4. For many other heuristics, it provides better performance. This is most likely because Potential Gain Calculation Method 3 and 4 weigh less on potential gain as remaining budget gets smaller, therefore already accounting for potential gains which are not likely to be ever realized. On the other hand, for individual heuristics in individual experiments, tuning $MinPGR$ can also result in better performance. For instance, TABU-PG_3_4_0.1 obtains a performance value of 179.8 while TABU-PG_3_4_0 obtains 176.8 in Experiment 4. Note that TABU-PG_3_4_0.1 is the second best performing heuristic for this experiment.

In Experiment 8, there exists several TABU-PG heuristics which performs more than 4000% better than the worst TABU-PG heuristic. This is due to the tipping points in the given network. Better heuristics reach the tipping point before others which results in very large performance improvement in terms of final spread. The impact of tipping points in Experiment 8 can also be seen in Figure 4.

Table 6 shows performances of different methods of $NSMthd$, $PGCMthd$, and $MinPGR$ in terms of average final influence spreads. Averages are calculated over 39 TABU-PG heuristics except for Experiment 7 where experiments with different $MinPGR$ values have not been performed. Experiment 8 is not included since heuristics in experiments are run with $TopMltp$ 10 instead of Inf. For each experiment, best performing methods on average are shown with bold font. All values are as percentages.

Node Selection Method 1 is the most naive method which does not account for budget or efficiency but only for gain. Hence, it is outperformed by Method 2 which maximizes the efficiency before gain. Method 3 which is introduced in this paper performs better than both methods on average. The intuition behind Method 3 is that efficiency is indeed more important than immediate gain, however it could be the case that selecting the node with highest gain

Table 6: Method Performances in each Experiment (%)

	$NSMthd$			$PGCMthd$				$MinPGR$			
	1	**2**	**3**	**1**	**2**	**3**	**4**	**0**	**0.05**	**0.1**	**0.2**
E1	103.6	114.7	**115.3**	109.2	110.1	111.6	**112.3**	111.2	**111.7**	111.4	111.1
E2	103.4	**108.4**	108.3	102.7	104.4	**108.5**	108.3	107.0	**107.1**	107.0	106.9
E3	108.4	**121.6**	121.6	**119.5**	113.2	119.2	118.7	116.5	116.7	117.1	**117.8**
E4	163.0	167.2	**173.2**	131.7	170.6	170.9	**171.0**	**172.6**	169.8	170.9	170.0
E5	103.1	105.2	**105.8**	103.8	103.7	105.0	**105.6**	104.7	104.6	**104.8**	105.0
E6	102.8	108.5	**108.8**	100.7	106.5	106.7	**108.4**	**107.4**	107.3	107.2	106.8
E7	116.7	**177.8**	177.7	**165.4**	139.6	164.0	160.8	-	-	-	-
σ	20.4	28.1	29.1	21.5	23.2	26.3	25.4	23.9	22.8	23.2	22.9
μ	114.4	129.1	**130.1**	119	121.2	**126.6**	126.4	**119.9**	119.5	119.7	119.6

among the top efficient nodes could be more effective. Experiments supported this argument.

Potential Gain Calculation Method 3 and 4 which are introduced in this paper outperform Method 1 and 2. Method 1 is the most naive method which do not account for potential gains altogether, thus it is outperformed by Method 2. Method 2 emphasizes on potential gains. However, investing in future potential gains when there is only a limited budget left is not wise. The methods introduced in this paper consider that and dynamically change the weight between actual gain and potential gain, hence are able to perform better than other methods.

A meaningful and consistent difference between performances of different Minimum Potential Gain Ratios is not observed on average. It suggests that the role $MinPGR$ is supposed to play is already taken care of by the methods we introduced in Node Selection and Potential Gain Calculation. However, tuning $MinPGR$ is able to produce better performances in individual cases.

Results of Scaling Methods

A single TABU-PG heuristic takes a few minutes for experiments on Epinions, approximately half an hour for experiments on Academia, less than half a minute for experiments on Inploid, and several hours for experiments on Pokec. It shows that as the network gets larger, the runtime increases at a faster pace. In addition, the distribution of threshold values and influence weights affect

Table 7: Impact of Scaling Methods on Performance (%)

ExH	**1_I**	**1_20**	**1_10**	**5_I**	**5_20**	**5_10**	**25_I**	**25_20**	**25_10**	**I_I**
1xM	100.0	101.9	102.9	93.9	99.9	101.0	86.9	92.0	99.5	94.4
1xB	100.0	100.4	100.4	92.8	97.7	99.4	82.5	94.8	93.7	92.2
2xM	100.0	98.5	96.9	98.1	97.4	96.6	94.6	96.4	95.9	91.9
2xB	100.0	98.8	97.3	98.1	97.6	96.8	93.1	97.2	96.1	87.3
3xM	100.0	99.5	99.5	99.1	99.4	99.4	98.6	99.2	99.3	98.9
3xB	100.0	99.8	99.8	99.8	99.8	99.8	99.8	100.0	99.9	95.7
4xM	100.0	96.8	93.2	90.6	91.3	91.3	70.3	85.4	85.1	16.6
4xB	100.0	93.7	86.0	83.7	89.2	85.0	70.0	77.4	78.0	15.8
5xM	100.0	98.9	98.2	99.2	97.6	97.7	93.9	95.2	95.3	89.1
5xB	100.0	100.0	100.0	95.9	97.4	98.0	90.3	91.7	94.5	90.5
6xM	100.0	100.2	99.3	92.2	94.4	95.0	89.6	90.2	91.7	79.9
6xB	100.0	99.4	97.9	94.5	95.5	94.4	89.0	87.1	87.9	77.6
7xM	100.0	96.5	95.5	95.3	95.5	95.3	91.8	94.1	94.0	84.0
7xB	100.0	98.2	98.0	98.5	98.1	97.7	97.0	96.9	96.6	72.4
8x-	100.0	96.6	95.7	98.4	96.4	95.0	95.5	94.6	6.5	0.8
σ	**0.0**	**2.0**	**3.9**	**4.2**	**3.0**	**4.0**	**8.9**	**5.9**	**5.9**	**26.0**
μ	**100.0**	**98.8**	**97.5**	**95.1**	**96.5**	**96.2**	**89.1**	**92.7**	**93.4**	**77.6**

the runtime since the number of activated nodes depends on those values; and status of nodes determines whether some calculations are done or skipped. As a result, the heuristic requires less than 3 hours in Experiment 7 whereas it requires more than 10 hours in Experiment 8.

Table 7 illustrates the impact of scaling methods over different heuristics in different experiments in terms of spread performance. The first column specifies the experiment and heuristic. For instance, 4xM is the median performing TABU-PG heuristic in Experiment 4 and 5xB is the best performing TABU-PG heuristic in Experiment 5. Only one heuristic is investigated for Experiment 8 and that heuristic is neither the best nor the median performing heuristic, thus it is specified as such in this table. Inf is further shortened as I. For example, I_I represents Inf_Inf for scaling methods. Heuristics with the default values for scaling methods are set as 100% while others are assigned values based on their relative performances.

Setting $NumBefReCalc$ to Inf would not be relevant to Potential Gain Calculation Method 3 and 4, however it is still included in the table for heuristics which employ Method 3 or 4. These two methods assign value to $PGMltp$

Table 8: Impact of Scaling Methods on Runtime (%)

ExH	**1_I**	**1_20**	**1_10**	**5_I**	**5_20**	**5_10**	**25_I**	**25_20**	**25_10**	**I_I**
1xM	100.0	21.7	14.5	65.5	13.7	9.9	28.1	8.0	7.1	5.3
1xB	100.0	19.4	13.3	62.2	12.5	8.6	25.9	7.6	6.6	4.8
2xM	100.0	20.6	17.4	67.3	17.1	15.5	36.1	16.4	14.8	12.1
2xB	100.0	18.9	17.2	53.9	20.0	17.1	55.7	15.9	14.9	19.1
3xM	100.0	11.5	10.8	63.7	14.5	8.9	34.0	9.1	8.6	6.5
3xB	100.0	16.6	13.6	82.1	19.3	12.0	43.0	12.1	11.4	8.4
4xM	100.0	8.1	6.8	70.2	8.0	6.8	38.9	6.9	6.3	4.3
4xB	100.0	8.0	7.1	69.6	7.8	7.0	38.7	7.1	6.7	5.3
5xM	100.0	23.1	15.4	53.8	15.4	15.4	23.1	7.7	7.7	7.7
5xB	100.0	31.3	20.5	63.0	22.3	17.0	41.2	15.6	13.3	7.8
6xm	100.0	27.3	18.2	63.6	27.3	18.2	36.4	18.2	18.2	9.1
6xB	100.0	27.3	18.2	54.5	27.3	18.2	36.4	18.2	18.2	9.1
7xM	100.0	41.7	38.1	77.1	36.0	33.7	73.3	35.3	35.2	22.4
7xB	100.0	38.4	42.7	76.2	40.0	38.2	67.5	39.0	38.5	26.6
8x-	100.0	11.9	10.0	93.5	11.4	10.2	55.7	15.8	10.6	7.2
σ	**0.0**	**9.8**	**9.9**	**8.3**	**9.3**	**9.0**	**14.2**	**9.8**	**9.9**	**6.8**
μ	**100.0**	**22.4**	**18.1**	**65.9**	**20.1**	**16.2**	**41.3**	**15.5**	**14.8**	**10.6**

according to the remaining budget whereas $NumBefReCalc = Inf$ selects all nodes at once in the first iteration instead of spending budget overtime. Therefore, Potential Gain Methods 3 and 4 is equivalent to Method 2 whenever $NumBefReCalc$ is set to Inf.

It can be depicted from Table 7, counter-intuitively at first, limiting calculations to top nodes by decreasing the value of *TopMltp* usually improves the total profit while also significantly reducing the runtime; in the case when $NumBefReCalc$ is a large value. For instance, on average, there is 4% points increase in total profit when candidate seed is limited by *TopMltp* being 10 or 20, given that $NumBefReCalc$ is set as 25. An explanatory idea for why this is the case is as follows. When large number of nodes are selected at once, there is almost always a decrease in spread performance. As the value of $NumBefReCalc$ grows, the decrease in spread performance becomes larger. When candidate pool is limited by setting *TopMltp* to values which are small enough but not too small, the decrease in performance due to the larger value of $NumBefReCalc$ is limited by the use of *TopMltp*.

Table 8 illustrates the impact of scaling methods over different heuristics in different experiments in terms of runtime. The columns and rows are same as Table 7, but displaying runtimes instead of total profit. The results show that the runtimes can be reduced to up to 5% of what it originally takes. Limiting the candidate pool to top nodes reduces the runtime to one fifth on average. Selecting 5 or 25 nodes instead of 1 at a time reduces the runtime by 34% and 59% on average. Combining these two methods can further reduce the runtime.

The scaling methods we created, $NumBefReCalc$ and $TopMltp$, enables TABU-PG to run on very large networks in reasonable amounts of time. Especially, limiting the candidate pool for seed nodes by assigning an appropriate value to $TopMltp$ significantly reduces the runtime while nearly matching the same influence spread. The runtime can further be reduced by employing $NumBefReCalc$ by sacrificing a little more influence spread. However, $NumBefReCalc$ should not be set to values which are too large since it might cause a dramatic decrease in performance especially in the cases where there exist tipping points.

5. Conclusion

In this paper, we defined the new Targeted and Budgeted Influence Maximization Problem under Deterministic LTM. We extended the original Influence Maximization Problem by allowing different nodes to carry different cost and return values under a Deterministic LTM. This makes it possible to model different real-world Influence Maximization problems depending on how the return values are generated; assigning values based on estimated profits would make it a profit maximization problem whereas assigning values based on distances to the target location would make it a location-based marketing problem. As a solution to the defined problem, we developed Targeted and Budgeted Potential Greedy (TABU-PG) algorithm. The term *Targeted* in TABU-PG is intended to cover all such problems.

The idea behind TABU-PG algorithm is to invest on potential future gains instead of accounting for only immediate gains (i.e., actual gains). In order to obtain a better return on such investments, we equip TABU-PG with methods which dynamically control such investments by employing a set of potential gain calculation procedures. One method discounts the potential gains if the partially influenced node is only very slightly influenced, hence not likely to be activated in future iterations. Another method dynamically controls the weights of potential gains in comparison to actual gains. As the remaining budget gets closer to zero, the emphasize given on potential gains is reduced in favour of actual gains. This is because, the chances of reaping those potential gains decreases as the budget is spent over time. The algorithm supplies different methods for different operations. Thus, each combination of method selection results in a distinct TABU-PG heuristic.

Most of data is synthetically generated for Influence Maximization problems in the literature. We propose new techniques for generating influence weights, thresholds, profits and costs. Our influence weight generation method results in a tipping point phenomenon present in many real-life diffusion dynamics. Computational experiments demonstrated that even the worst TABU-PG heuristic performs better than the best benchmark heuristic excluding some rare cases where an average TABU-PG heuristic still performs better than all benchmarks.

TABU-PG is readily designed to minimize the number of calculations by detecting redundant calculations and skipping them. Further to that, we proposed additional novel methods which significantly improve runtime by providing a trade-off between runtime and spread performance. The reduction in spread performance is shown to be often very slight. Utilizing these scalability methods, TABU-PG is able to run on very large networks in a reasonable amount of time.

TABU-PG has a modular structure with clearly distinguished alternative subprocesses. For example, four different potential gain calculation methods and three different node selection methods are presented in this study. Future research efforts, then, might be directed towards designing alternative subprocesses which provide better performance in terms of run time or obtained influence spread. Moreover, TABU-PG is designed to work with any set of parameter values. Another research direction would be to focus on influence models and developing learning techniques, together with actual diffusion data, to better estimate parameters such as influence weights and thresholds for any given setting. As such values become available with greater confidence, TABU-PG will further continue to serve as an effective decision support tool regarding marketing campaigns in social networks.

Acknowledgment

F. Gürsoy is supported by the Scientific and Technological Research Council of Turkey (TUBITAK) under 2210-A Program.

References

[1] M. Gladwell, The tipping point: how little things can make a big difference, Little Brown, New York, NY, USA, 2006.

[2] D. Kahneman, Thinking, fast and slow, Macmillan, New York, NY, USA, 2011.

[3] E. M. Rogers, Diffusion of innovations, Free Press of Glencoe, New York, NY, USA, 1962.

[4] R. Ferguson, Word of mouth and viral marketing: taking the temperature of the hottest trends in marketing, Journal of Consumer Marketing 25 (3) (2008) 179–182.

[5] P. Domingos, M. Richardson, Mining the network value of customers, in: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2001, pp. 57–66.

[6] E. Even-Dar, A. Shapira, A note on maximizing the spread of influence in social networks, Internet and Network Economics (2007) 281–286.

[7] D. Kempe, J. Kleinberg, É. Tardos, Maximizing the spread of influence through a social network, in: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, Washington, DC, USA, 2003, pp. 137–146.

[8] A. Goyal, W. Lu, L. V. S. Lakshmanan, Simpath: An efficient algorithm for influence maximization under the linear threshold model, in: Data Mining (ICDM), 2011 IEEE 11th International Conference on, IEEE, Vancouver, Canada, 2011, pp. 211–220.

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

[10] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, N. Glance, Cost-effective outbreak detection in networks, in: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, San Jose, CA, USA, 2007, pp. 420–429.

[11] A. Goyal, W. Lu, L. V. S. Lakshmanan, Celf++: optimizing the greedy algorithm for influence maximization in social networks, in: Proceedings of the 20th International Conference Companion on World Wide Web, ACM, Hyderabad, India, 2011, pp. 47–48.

[12] C. Zhou, P. Zhang, J. Guo, X. Zhu, L. Guo, Ublf: An upper bound based approach to discover influential nodes in social networks, in: Data Mining (ICDM), 2013 IEEE 13th International Conference on, IEEE, Dallas, TX, USA, 2013, pp. 907–916.

[13] W. Chen, Y. Wang, S. Yang, Efficient influence maximization in social networks, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, Paris, France, 2009, pp. 199–208.

[14] W. Chen, C. Wang, Y. Wang, Scalable influence maximization for prevalent viral marketing in large-scale social networks, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, Washington, DC, USA, 2010, pp. 1029–1038.

[15] F. Li, C. Li, M. Shan, Labeled influence maximization in social networks for target marketing, in: Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third International Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on, IEEE, Boston, MA, USA, 2011, pp. 560–563.

[16] W. Lu, L. V. S. Lakshmanan, Profit maximization over social networks, in: Data Mining (ICDM), 2012 IEEE 12th International Conference on, IEEE, Brussels, Belgium, 2012, pp. 479–488.

[17] Y. Li, D. Zhang, K. Tan, Real-time targeted influence maximization for online advertisements, Proceedings of the VLDB Endowment 8 (10) (2015) 1070–1081.

[18] J. Lee, C. Chung, A query approach for influence maximization on specific users in social networks, IEEE Transactions on knowledge and Data engineering 27 (2) (2015) 340–353.

[19] C. Song, W. Hsu, M. L. Lee, Targeted influence maximization in social networks, in: Proceedings of the 25th ACM International Conference on Information and Knowledge Management, ACM, Indianapolis, IN, USA, 2016, pp. 1683–1692.

[20] H. Nguyen, R. Zheng, On budgeted influence maximization in social networks, IEEE Journal on Selected Areas in Communications 31 (6) (2013) 1084–1094.

[21] N. Du, Y. Liang, M. F. Balcan, L. Song, Budgeted influence maximization for multiple products, arXiv preprint arXiv:1312.2164.

[22] Y. Singer, How to win friends and influence people, truthfully: influence maximization mechanisms for social networks, in: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, ACM, Seattle, WA, USA, 2012, pp. 733–742.

[23] Z. Lu, W. Zhang, W. Wu, J. Kim, B. Fu, The complexity of influence maximization problem in the deterministic linear threshold model, Journal of Combinatorial Optimization 24 (3) (2012) 374–378.

[24] Z. Lu, W. Zhang, W. Wu, B. Fu, D. Du, Approximation and inapproximation for the influence maximization problem in social networks under deterministic linear threshold model, in: Distributed Computing Systems Workshops (ICDCSW), 2011 31st International Conference on, IEEE, Minneapolis, MN, USA, 2011, pp. 160–165.

[25] F. Zou, Z. Zhang, W. Wu, Latency-bounded minimum influential node selection in social networks, in: International Conference on Wireless Algorithms, Systems, and Applications, Boston, MA, USA, 2009, pp. 519–526.

[26] X. Zhu, J. Yu, W. Lee, D. Kim, S. Shan, D. Du, New dominating sets in social networks, Journal of Global Optimization 48 (4) (2010) 633–642.

[27] F. Wang, H. Du, E. Camacho, K. Xu, W. Lee, Y. Shi, S. Shan, On positive influence dominating sets in social networks, Theoretical Computer Science 412 (3) (2011) 265–269.

[28] W. Zhang, W. Wu, F. Wang, K. Xu, Positive influence dominating sets in power-law graphs, Social Network Analysis and Mining 2 (1) (2012) 31–37.

[29] G. Askalidis, R. A. Berry, V. G. Subramanian, Explaining snapshots of network diffusions: Structural and hardness results, in: International Computing and Combinatorics Conference, Springer, Atlanta, GA, USA, 2014, pp. 616–625.

[30] D. Acemoglu, A. Ozdaglar, E. Yildiz, Diffusion of innovations in social networks, in: Decision and Control and European Control Conference (CDC-

ECC), 2011 50th IEEE Conference on, IEEE, Orlando, FL, USA, 2011, pp. 2329–2334.

[31] R. Xu, An lp norm relaxation approach to positive influence maximization in social network under the deterministic linear threshold model, in: International Workshop on Algorithms and Models for the Web-Graph, Springer, Montreal, Canada, 2013, pp. 144–155.

[32] A. Swaminathan, An algorithm for influence maximization and target set selection for the deterministic linear threshold model, Master's thesis, Virginia Polytechnic Institute and State University (2014).

[33] D. Gunnec, S. Raghavan, Integrating social network effects in the share-of-choice problem, Decision Sciences 48 (6) (2017) 1098–1131.

[34] J. Leskovec, A. Kreyl, SNAP Datasets: Stanford large network dataset collection, `http://snap.stanford.edu/Data` (Jun. 2014).

[35] M. Richardson, R. Agrawal, P. Domingos, Trust management for the semantic web, The Semantic Web-ISWC 2003 (2003) 351–368.

[36] M. Fire, L. Tenenboim, O. Lesser, R. Puzis, L. Rokach, Y. Elovici, Link prediction in social networks using computationally efficient topological features, in: Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on, IEEE, Boston, MA, USA, 2011, pp. 73–80.

[37] M. Fire, L. Tenenboim-Chekina, R. Puzis, O. Lesser, L. Rokach, Y. Elovici, Computationally efficient link prediction in a variety of social networks, ACM Transactions on Intelligent Systems and Technology (TIST) 5 (1) (2013) 10.

[38] L. Takac, M. Zabovsky, Data analysis in public social networks, in: International Scientific Conference and International Workshop Present Day Trends of Innovations, Vol. 1, 2012.

[39] S. Aral, P. S. Dhillon, Social influence maximization under empirical influence models, Nature Human Behaviour (2018) 1.

[40] C. P. Robert, Simulation of truncated normal variables, Statistics and Computing 5 (2) (1995) 121–125.

[41] L. Page, S. Brin, R. Motwani, T. Winograd, The pagerank citation ranking: bringing order to the web., Tech. rep., Stanford InfoLab (1999).

[42] J. Kleinberg, Authoritative sources in a hyperlinked environment, Journal of the ACM (JACM) 46 (5) (1999) 604–632.

[43] P. Bonacich, Power and centrality: a family of measures, American Journal of Sociology 92 (5) (1987) 1170–1182.