An exact method for influence maximization based on deterministic linear threshold model

Eszter Julianna Csókás[1] and Tamás Vinkó[1]

[1]Department of Computational Optimization, University of Szeged, Hungary.

Contributing authors: csokas@inf.u-szeged.hu; tvinko@inf.u-szeged.hu;

Abstract

Influence maximization (IM) is a challenging combinatorial optimization problem on (social) networks given a diffusion model and limited choice for initial seed nodes. In a recent paper by Keskin and Güler (2018) an integer programming formalization of IM using the so-called deterministic linear threshold diffusion model was proposed. In fact, it is a special 0-1 linear program in which the objective is to maximize influence while minimizing the diffusion time. In this paper, by rigorous analysis, we show that the proposed algorithm can get stuck in locally optimal solution or cannot even start on certain input graphs. The identified problems are resolved by introducing further constraints which then leads to a correct algorithmic solution. Benchmarking results are shown to demonstrate the efficiency of the proposed method.

Keywords: Influence maximization, deterministic linear threshold, integer linear programming

1 Definitions

Influence maximization. Let $G=(V,E,W)$ be a directed weighted graph, where V is the set of nodes, E is the set of edges and $W: E \to \mathbf{R}_+$ is a non-negative weight function. Influence maximization (IM) is a combinatorial optimization problem in which, given a weighted directed graph G, a diffusion (or spreading) model, and an integer $k \geq 1$, it is required to identify the so-called seed nodes $v_1, \ldots, v_k \in V$ which can make the largest influence in the network (Kempe, Kleinberg, & Tardos, 2003).

In the following it is assumed that $n = |V|$. For a given node $j \in V$ the set of in-neighbors is denoted by $N(j)$. For an integer $0 < k \leq n$, the function $\sigma(S)$ determines that choosing a node set $S \subset V$ of size k as seed set, how many nodes will be influenced by executing the spreading model. Formally, the optimization problem can be described as

$$\max_{S \subset V, |S|=k} \sigma(S).$$

Kempe et al. (2003) investigated the influence maximization problem using spreading models with stochastic parameters. Hence, $\sigma(S)$ stands for the expected value of the number of influenced nodes.

The influenced and uninfluenced nodes will also be called as active and inactive nodes, respectively.

Linear threshold model. Several diffusion models have been proposed in the literature. Apart from the linear threshold model (Granovetter, 1978), on which our paper is based, the most frequently used ones are the independent cascade model (Goldenberg, Libai, & Muller, 2001), triggering model (Kempe et al., 2003) and time-aware model (Liu, Cong, Xu, & Zeng, 2012).

Let $b_{i,j} \in (0,1)$ be the edge weight between node i and j, $\theta_i \in (0,1]$ be the threshold of node i, and set $\hat{N}(i)$ be the already influenced in-neighbors of node i.

The linear threshold (LT) model starts from $t = 1$ (where $t \in \mathbb{N}$), and iteratively does the following steps by increasing the value of t:

Step 1 Let $0 < k \leq n$ be fixed, a seed set $\mathscr{V}_0$ containing k nodes, $t = 1$, $\mathscr{V}_1 = \emptyset$.
Step 2 For all $i \in V$ inactive nodes, if

$$\sum_{j \in \hat{N}(i)} b_{j,i} \geq \theta_i$$

holds, then put node i into the set $\mathscr{V}_t$, in which at the end of this step all the nodes are labeled as influenced.
Step 3 If $\mathscr{V}_t = \emptyset$ holds, meaning that it is not possible to make more nodes influenced, then STOP; the set of influenced nodes is $\mathscr{V} = \mathscr{V}_0 \cup \ldots \cup \mathscr{V}_{t-1}$ and $\sigma(\mathscr{V}_0) = |\mathscr{V}|$. Otherwise, let $t := t + 1$ and go back to Step 2.

The threshold value θ_i determines the influenceability of node i. According to the original paper of Kempe et al. (2003), it is arguably difficult to measure (e.g., in social networks) the value of this thresholds. Hence, the evaluation of the LT model is done by executing it R times and then the average influence value is taken; this is how the expected value of σ is obtained. In this case it can be shown that the function $\sigma(\cdot)$ has *submodularity* property, which has the important consequence that a greedy algorithm guarantees that

$$\sigma(S) \geq (1 - 1/e) \cdot \sigma(S^*)$$

holds for any seed set S, where S^* is the optimal seed set (Nemhauser, Wolsey, & Fisher, 1978).

Deterministic linear threshold model. In the deterministic LT (DLT) model all the θ_i threshold values are fixed. In the recent years, this model has been studied (Acemoglu, Ozdaglar, & Yildiz, 2011; Karampourniotis, Szymanski, & Korniss, 2019; Lu, Zhang, Wu, Fu, & Du, 2011; Lu, Zhang, Wu, Kim, & Fu, 2012; Xu, 2013). In fact, the original LT model by Granovetter (1978) is also deterministic. One of the most interesting fact about DLT that submodularity does not hold (Altarelli, Braunstein, Dall'Asta, & Zecchina, 2013), thus the greedy algorithm cannot be expected to be efficient. Our paper focuses on the DLT model.

2 Related works

Among the vast amount of scientific contributions related to the IM problem, the most relevant works to our paper are the ones using ILP models and/or based on the DLT diffusion model.

For influence *minimization* Yang, Giua, and Li (2017) gives an ILP model, which is another problem. Using the LT model Rosa and Giua (2013) gives such ILP formalism in which the aim is to determine the set of nodes which will never get influenced. This information might be used for solving the original problem. The already mentioned paper of Altarelli et al. (2013) considers the problem as constraint satisfaction and investigates the efficiency of belief propagation algorithm.

In Güney (2019) a binary integer program that approximates the IM using independent cascade diffusion model (which is different than the one used in this paper) by Monte Carlo sampling is developed together with a linear programming relaxation based method with a provable worst case bound.

The stochastic version of the IM problem was investigated by Wu and Küçükyavuz (2018). They developed a two-stage stochastic programming framework using delayed constraint-generation algorithm. The paper Kahr, Leitner, Ruthmair, and Sinnl (2021) focuses on competitive IM based on probabilistic independent cascade model in which the seed individuals of one entity is already known, while another entity wants to choose its seed set of individuals that triggers an influence cascade of maximum impact. An algorithmic framework based on a Benders decomposition is developed which enables to handle graphs with thousands of nodes and edges. Note that the full ILP model in Keskin and Güler (2018) also considers competition explicitly. Another reformulation based on Benders decomposition of the IM problem using probabilistic independent cascade model was developed in (Güney, Leitner, Ruthmair, & Sinnl, 2021).

Nannicini, Sartor, Traversi, and Calvo (2020) investigates a robust optimization problem using the DLT model. It is assumed that the nodes' thresholds and the edge weights can change within a certain domain. The problem of our paper is a special version of this general one. They construct such an ILP model in which the time parameter t does not play a role (in contrast to our work), moreover, the number of variables grow exponentially. The DLT model was also used in Chen, Pasiliao, and Boginski (2020) where an arc-based mixed-integer programming model has been developed for the so-called Least Cost Influence Maximization Problem and thus it is a different problem. A possible extension of the IM problem was defined in Gursoy and Gunnec (2018). Namely, the Targeted and Budgeted Influence Maximization problem under DLT model was developed, which allows different nodes to carry different cost and return values. This problem was investigated by using a scalable greedy approach.

In (Farzaneh, Masoud, & Heshaam, 2021) a new approach was presented called MLPR (matrix multiplication, linear programming, randomized rounding) with linear programming used as its core in order to solve the IM problem with LT model. Although the method was shown to be efficient both in running time and in the quality of the result, it does not have an approximation guarantee.

It was shown by Cheng, Kuo, and Zhou (2020) that the IM problem using LT diffusion model is equivalent to the so-called targeted immunization problem. A mixed-integer linear programming formalism was developed together with Benders decomposition approach. A much more general IM problem was introduced by Shunyu, Neng, and Jie (2022) to study the spread of infectious disease process. The considered model takes the cumulative effect of LT into account.

Michael, Markus, and Ivana (2022) studied three new variants of the competitive influence maximization problem (CIMP) which consideration of passive (viewing-only) nodes, node resistance, and customer choice behavior. For solving these problems a mixed-integer nonlinear programming model was proposed.

Obtaining realistic parameters, such as nodes' threshold values and edge weights for real-world graphs could be challenging. A mixed-integer linear programming model and an approximate method using artificial neural network have been proposed to learn the edge weights in the LTM for synthetic and real data by (Qiang, Pasiliao, & Zheng, 2019). Regarding the estimation of threshold values Talukder et al. (2019) comprehensively surveys the different threshold values used in various IM models and develops four threshold estimation models based on edge weight and degree distribution.

There are many papers about the usage of different network centrality metrics for identifying the seed nodes in IM. Due to the fact that in IM the underlying graph is not only weighted and directed, but also labeled, i.e., by the nodes' threshold values, the classical centrality metrics fail to provide high quality results. On the other hand, diffusion models such as DLT can serve as basis for introducing new centrality metrics, like it is done, e.g., in Riquelme, Gonzalez-Cantergiani, Molinero, and Serna (2018). Our ILP model is impractical for direct usage as centrality metrics. Nevertheless, thorough analysis of the solution matrix $\mathbf{x}$ could reveal certain properties of the network.

Finally, for a detailed overview on the IM problem and its definitions, computational complexities, heuristic solution methods the reader is referred to the survey by Li, Fan, Wang, and Tan (2018).

3 A model and a proposed algorithm

3.1 A 0-1 linear programming model

Two integer linear programming formulations of influence maximization based on the DLT model were recently proposed and studied in Keskin and Güler (2018). The first one, referred as *basic* model, includes a single party trying to find the initial seed nodes to maximize the spread of influence; while the second one, referred as *competition* model, extends the first one by introducing an enemy trying to spread its own influence. In the current paper we investigate the basic model of Keskin and Güler (2018), more precisely, not even the cost of selecting a seed node is taken into account.

The formulation of the basic model is a special 0-1 LP, in which $\mathbf{x} \in \{0,1\}^{n\times\mathscr{T}}$ is the decision variable, $n = |V|$, and the index $\mathscr{T} > 1$ is also part of the optimization problem. Hence, $\mathbf{x}$ is a binary matrix in which choosing the rows in the first column to be equal to 1 represents the selection of the seed nodes. This should be done in such a way that, given certain constraints dictated by DLT model, the sum of the last column is to be maximized.

Assuming that $\mathscr{T} > 1$ is a given integer constant, let $T = \{2,\ldots,\mathscr{T}\}$ be the set of time periods describing the diffusion process. Let integer $k > 0$ be the number of seed nodes to be selected. The set of in-neighbors of node i is denoted by $N(i)$.

In the following the binary LP formulation is given, inspired by the basic model of Keskin and Güler (2018), where the cost of selecting a seed node is equal to 1.

$$\max \sum_{i=1}^{n} x_{i,\mathscr{T}} \tag{1}$$

$$\sum_{i=1}^{n} x_{i,1} \leq k \tag{2}$$

$$\sum_{j\in N(i)} b_{j,i} x_{j,t-1} \geq \theta_i x_{i,t} \quad \forall (i \in V, t \in T) \tag{3}$$

$$\sum_{j\in N(i)} b_{j,i} x_{j,t-1} \leq \theta_i + x_{i,t} \quad \forall (i \in V, t \in T) \tag{4}$$

$$x_{i,t-1} \leq x_{i,t} \quad \forall (i \in V, t \in T) \tag{5}$$

$$\mathbf{x} \in \{0,1\}^{n\times\mathscr{T}} \tag{6}$$

In the objective function (1) the number of influenced nodes are maximized in the last time period. The constraint (2) limits the number of seed nodes to be selected initially. The constraint (3) guarantees that node i cannot be influenced at time period t if the total weighted in-degree from the already influenced neighbors is below the threshold value of node i. Furthermore, by constraint (4), if node i's threshold at time period t is exceeded by the weighted in-degree from the already influenced neighbors, then node i gets influenced. It is important to emphasize here that it is assumed that the sum of in-weights of nodes cannot exceed 1. The constraint (5) ensures that influenced nodes remain to be so in later time periods, whereas constraint (6) restricts the solution matrix to be binary.

The objective function in fact has the form

$$\min_{\mathscr{T}} \max \sum_{i=1}^{n} x_{i,\mathscr{T}}$$

and together with constraints (2) - (6) we have a bilevel optimization problem. It is shown that linear bilevel problems are strongly NP-hard (Hansen, Jaumard, & Savard, 1992).

The AMPL modeling language (Fourer, Gay, & Kernighan, 1993), which we used for implementation and numerical experiments (see Section 5), is not suitable for directly describing bilevel optimization models. That would require to have declarations as `var T; var x{n,T};` which is not supported. Hence, we need to consider and treat $\mathscr{T}$ as constant.

Remark 1 The globally optimal solution for the bilevel problem is when we have the maximal influence within the shortest diffusion time. This will be referred as $(\sigma^*, \mathscr{T}^*)$ in the following.

3.2 An iterative algorithm

The solution method for the bilevel optimization problem proposed in Keskin and Güler (2018) is shown in Algorithm 1.

Algorithm 1

```
Step 1  Start the iteration from $\mathscr{T} := 2$.
Step 2  Solve the optimization problem (1) - (6) with fixed $\mathscr{T}$.
Step 3  If $\mathbf{x}_{i,\mathscr{T}} = \mathbf{x}_{i,\mathscr{T}-1} \quad \forall(i \in V)$, i.e., the last two columns of $\mathbf{x}$ are the same then
STOP, the optimum is found. Otherwise, let $\mathscr{T} := \mathscr{T} + 1$ and go back to Step 2.
```

This iterative methods makes it possible to find the minimal $\mathscr{T}$ since it stops when further spreading of influence is not possible. Thus, the value of $\mathscr{T}^*$ is given by the loop variable $\mathscr{T}$.

4 Analysis

In this section a thorough analysis of Algorithm 1 proposed in Keskin and Güler (2018) and shown in Section 3.2 is given.

For a start, it turns out that the optimization problem (1)-(6) needs to be modified.

Proposition 1 *The constraint* (3) *needs to be replaced by*

$$\sum_{j \in N(i)} b_{j,i} x_{j,t-1} \geq \theta_i (x_{i,t} - x_{i,t-1}) \;\; \forall (i \in V, t \in T). \tag{7}$$

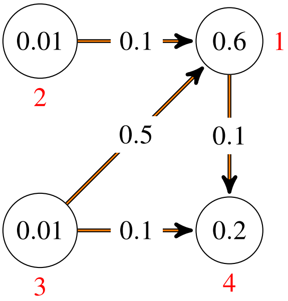

Fig. 1 Example graph to show the need of the new constraint (7)

Proof The model (1)-(6) dictates that the optimal seed nodes are those which have the maximal number of influenceable neighbors. However, by constraint (3) these seed nodes need also be influenced, which is only possible if these selected seed nodes form a set of size k in which the node's weighted in-degree is larger than its threshold. This cannot be held in general, thus it might happen that we obtain sub-optimal solution or it even becomes impossible to select seed nodes, and hence the influence spreading cannot be started.

On the other hand, by replacing constraint (3) with (7) all nodes could be selected as seed node, and thus the optimal solution could be found.

□

As an illustrative example, see the graph on Fig. 1. The labels of the nodes are indicated as red numbers. By constraint (3) the influence spreading cannot be started. The matrix $\mathbf{x}$ corresponding to the correct global optimum for this graph is

$$\mathbf{x} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

As it can be seen, the nodes represented by row 2 and row 3 are selected as seed nodes, which both have threshold value 0.01. These two nodes are not connected to each other, hence the model (1)-(6) is infeasible at time $\mathscr{T} = 2$.

Remark 2 Note that constraint (7) is equivalent to constraint (3) together with adding loop edges to all the nodes. However, it turns out that from the computational efficiency point of view using (7) directly is more beneficial.

In the following we show that Algorithm 1 can get stuck in locally optimal solution even if the newly added constraint (7) is taken into account.

Proposition 2 *For the optimization problem* (1), (2), (4) - (7), *there is a graph for which*

$$(\sigma, \mathscr{T}) = (\sigma, \mathscr{T}+1) \quad \textit{and} \quad (\sigma, \mathscr{T}+1) < (\sigma, \mathscr{T}+2).$$

Proof Such a graph is shown on Fig. 2, the solution matrix for $\mathscr{T} = 3$ is

$$\mathbf{x} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

By choosing nodes $\{1,4\}$ as seed nodes, for $\mathscr{T} = 2$ we have $\sigma = 4$ and the algorithm cannot increase the number of influenced nodes from $\mathscr{T} = 2$ to $\mathscr{T} = 3$ because the graph structure does not allow to increase the number of active nodes by changing the seed nodes. Moreover, any other seed nodes result less or equal number of active nodes at $\mathscr{T} = 3$. On the other hand, it can increase the number of active nodes at $\mathscr{T} = 4$ by changing the seed nodes to $\{1,3\}$ and on this case, all of the nodes will be active. □

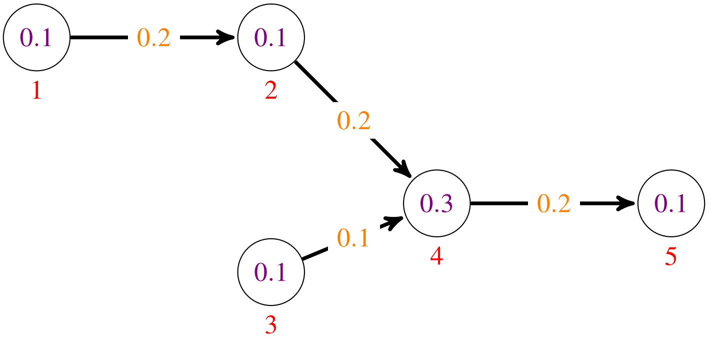

Fig. 2 Example graph for Proposition 2

We conclude that an extension of the optimization model (1), (2), (4) - (7) is needed in order to have a strategy about when to stop the iterative algorithm to be sure that it indeed reached the globally optimal solution. At that end, the following constraint is added:

$$\sum_{i=1}^{n} x_{i,\mathscr{T}-1} + 1 \leq \sum_{i=1}^{n} x_{i,\mathscr{T}}. \tag{8}$$

The purpose of constraint (8) is to force that for a given $\mathscr{T}$, the last step of the diffusion must have at least one more influenced node than in the previous step. We can thus guarantee no repetition in the last two columns of matrix $\mathbf{x}$.

Remark 3 Note that adding constraint (8) to the binary ILP model is in a direct contradiction to Algorithm 1, thus from now on we are into developing an alternative version.

Remark 4 The constraint (8) can be easily extended to any two consecutive columns in matrix $\mathbf{x}$. The overall performance of that version is discussed in Section 5.

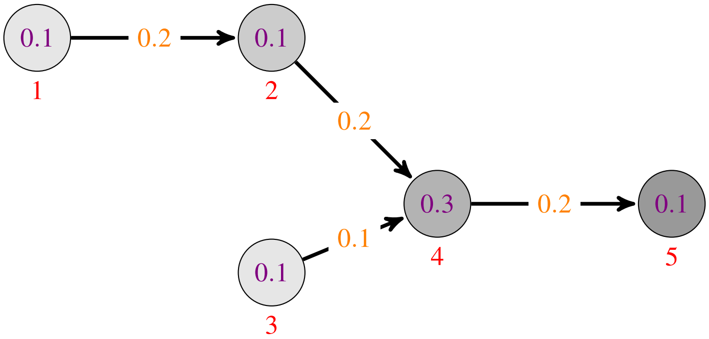

Fig. 3 Solution of example graph for Proposition 2 with (8)

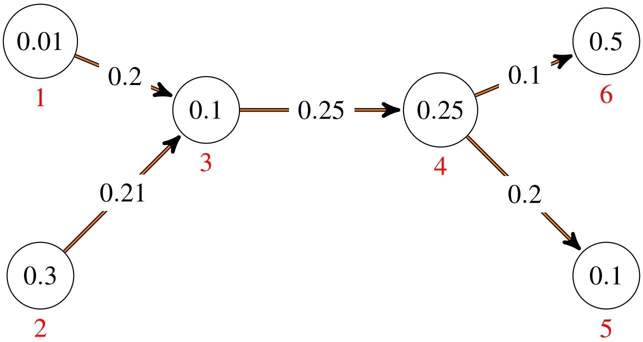

Fig. 4 Example graph for Proposition 3

Remark 5 On the Fig. 3 the graph is same than Fig. 2. By choosing $\{1,3\}$ as seed nodes, all of the nodes are active at $\mathscr{T} = 4$ with using constraint (8). The gray shading of the nodes indicate the time when the node gets activated.

The following proposition claims that although constraint (8) guarantees no repetition in the last two columns of $\mathbf{x}$, we can obtain such result in which column duplication appears inside the solution matrix.

Proposition 3 *For increasing $\mathscr{T}$ values the solutions of* (1), (2), (4) - (8) *do not necessary form a monotonically increasing sequence. Moreover, it can also happen that repetition occurs for consecutive columns in matrix* $\mathbf{x}$.

Proof As an example we refer to the graph shown in Fig. 4. The global optimum needs $\mathscr{T} = 4$ diffusion steps. By allowing further iteration steps to be taken by the algorithm, we expect to obtain an infeasible solution. However, the solution matrix for $\mathscr{T} = 5$ is

$$\mathbf{x} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

which contains repetition in its second and third column, hence lengthening the spreading up to $\mathscr{T} = 5$. This column repetition could go on forever. Note that this solution matrix is not representing the global optimum simply because the minimum time is $\mathscr{T}^* = 4$. This phenomenon is caused by constraints (4) and (7), which is explained in the graph shown on Fig. 4. According to matrix $\mathbf{x}$ reaching the node with the threshold value 0.25 could be delayed, thus we examine that case. Before reaching that node, constraint (7) always gets trivially satisfied, given the fact that its right hand side equals to 0. When the neighbor of the node in question is already activated, then on the left hand side of the constraint (4) the weight of the incoming edge appears, while we have either 0 or the threshold of the node on its right hand side. Since both values satisfy constraint (4), the algorithm allows to have the activation of the node after getting the global optimum. □

The example above shows that adding (8) to the ILP model can cause infinite loop in the iterative approach. It is caused by the possibility of column repetition inside the matrix $\mathbf{x}$, as it is explained in the proof of Proposition 3. This can be avoided by changing constraint (4) into

$$\sum_{j \in N(i)} b_{j,i} x_{j,t-1} \leq \theta_i + x_{i,t} - \varepsilon \qquad \forall (i \in V, t \in T) \tag{9}$$

where $\varepsilon > 0$ is a small constant to make sure that the node is activated when the sum of the edge weight of the already influenced in-neighbors of node is equal to the threshold. The choice for ε is discussed in Section 5.

The following proposition claims that adding constraint (8) to the ILP model does not prune the globally optimal solution.

Proposition 4 *The globally optimal solution of* (1), (2), (5) - (7), (9) *satisfies constraint* (8) *as well.*

Proof The globally optimal solution $(\sigma^*, \mathscr{T}^*)$ of (1), (2), (5) - (7), (9) cannot contain repetitions in the last $m > 1$ columns in its matrix $\mathbf{x}$ because in that case $(\sigma^*, \mathscr{T}^* - m + 1)$ would be a better solution. Due to constraint (9) matrix $\mathbf{x}$ cannot contain more column repetitions. Thus constraint (8) is satisfied. □

In addition to the previous proposition, it can also be shown that adding constraint (8) to the ILP model does not change the globally optimal solution.

Proposition 5 *The diffusion value $\mathscr{T}^*$ and influence value σ^* corresponding to the globally optimal solution of* (1)*,* (2)*,* (5) *-* (7)*,* (9) *are respectively the same as the values $\mathscr{T}^{**}$ and σ^{**} corresponding to the global optimum of* (1)*,* (2)*,* (5) *-* (9)*.*

Proof By introducing the constraint (8) such an optimization problem is obtained in which the value of $\mathscr{T}$ cannot be increased forever: at a certain point it gets an infeasible solution.

Firstly, let us see if $\sigma^{**} = \sigma^*$ holds. We have to check two cases.

- Assume that $\sigma^* < \sigma^{**}$. By dropping the constraint (8), we have a better solution for the problem (1), (2), (5) - (7), (9), which is not possible, since $(\sigma^*, \mathscr{T}^*)$ is the globally optimal solution.
- Assume that $\sigma^* > \sigma^{**}$. By Proposition 4 we know that a solution of (1), (2), (5) - (7), (9) also satisfies constraint (8) as well. Thus σ^* would be a better solution for the problem (1), (2), (5) - (9), which is not possible as σ^{**} is maximal.

We have contradictions for both cases, thus $\sigma^* = \sigma^{**}$.

Secondly, we check whether $\mathscr{T}^{**} = \mathscr{T}^*$ holds. We have to check again two cases.

- Assume that $\mathscr{T}^* < \mathscr{T}^{**}$. By Proposition 4 this is not possible as constraint (8) would not be satisfied.
- Assume $\mathscr{T}^* > \mathscr{T}^{**}$. We know that σ^* is global optimum for the problem (1), (2), (5) - (9) as well. This solution cannot be found with smaller amount of iteration steps under the constraints (8).

We have contradictions again, thus $\mathscr{T}^* = \mathscr{T}^{**}$. □

Now we need to find stopping conditions to the iterative procedure. Clearly, one of them is when all nodes are influenced. The other one is when the model becomes infeasible.

Proposition 6 *If the problem* (1)*,* (2)*,* (5) *-* (9) *becomes infeasible for a given $\mathscr{T}$ value, then it remains to be infeasible for the further iteration steps as well.*

Proof We show that if a solution is feasible then it was so in earlier iteration steps.

- $\mathscr{T} = 2$: The algorithm was able to do the first iteration, thus it could select the seed nodes. Hence, it has a feasible solution at $\mathscr{T} = 1$.
- $\mathscr{T} = 3$: By constraint (8) the last two columns cannot be the same. The first two columns of matrix $\mathbf{x}$ are certainly feasible, the corresponding nodes can be reach within this time frame.
- $\mathscr{T} = m$: In case we remove the mth column (where $m > 3$), a feasible solution is obtained since the nodes in the $(m-1)$th column could be reached and activated in $\mathscr{T} - 1$ steps. It is important to see that this solution is not necessarily globally optimal for all $t \in T$.

□

Finally, implicated by Proposition 5 and 6 we have the following consequence.

Corollary 7 *The problem* (1)*,* (2)*,* (5) *-* (9) *is feasible in the iteration steps $2, \ldots, \mathscr{T}^*$, i.e., before finding the global optimum.*

A correct model and iterative algorithm. Based on the analysis above, the correct iterative algorithm to find the globally optimal solution of the influence maximization problem under deterministic linear threshold diffusion model is given in Algorithm 2.

Algorithm 2

Step 1 Start the iteration with $\mathscr{T} := 2$.
Step 2 Solve the problem defined by the set of equations $\{(1), (2), (5), (6), (7), (8), (9)\}$ for the diffusion time value $\mathscr{T}$.
Step 3 If the solution becomes infeasible or all the nodes are influenced then STOP, the global optimum is found. Otherwise, let $\mathscr{T} = \mathscr{T} + 1$ and go back to Step 2.

5 Numerical experiments

5.1 Computational environment

The implementation of all the investigated ILP models were done in AMPL (Fourer et al., 1993). For the numerical experiments the solver Gurobi 9.5 was used with the non-default options: `threads=1 lpmethod=0 cuts=0 mipgapabs=1e-2`, which, compared to the default options, turned out to be much more efficient for these particular models. The computer used had Intel Xeon CPU E5-2660 at 2.00GHz with 64G memory running Ubuntu Linux 18.04.5.

5.2 Test graphs

For benchmarking the proposed algorithm some random graphs were generated. Two types of random graphs were used: Watts-Strogatz (WS) small-world graphs (Watts & Strogatz, 1998) and so-called LFR graphs with prescribed community structures (Lancichinetti, Fortunato, & Radicchi, 2008). For both types $5-5$ graph instances were generated.

WS graphs. These graphs were generated by using the package `R/igraph`. The parameters were:

- number of nodes is 60,
- number of neighbors in the starting graphs are $s = 4, 8$ and 12,
- and the rewiring probabilities (i.e., the probability of changing a directed edge $(v_1, v_2) \in E$ into a new edge (v_1, v_3), where $v_1 \neq v_2 \neq v_3$) are $\beta = 0.1$ and 0.3.

The `mutual` parameter was used which makes the graphs directed by doubling the undirected edges. Then 45% of randomly selected edges got removed. The edge weights were assigned as follows.

- First, for each edge a uniform at random number were generated in the interval $[0, 1]$.
- Nodes with larger than 1 in-weights were normalized to 1.

- Moreover, we applied a multiplication with a factor r_w which was a uniform at random number in the interval $[0.6,1]$.

The threshold values of the nodes were generated uniform at random in the interval $[0.15,0.4]$.

Using this particular procedure we were able to find such WS graphs on which the greedy algorithm found suboptimal solutions.

LFR graphs. These graphs were generated by the code from Lancichinetti et al. (2008), obtaining weighted directed graphs with community structure (thus, resembling social networks). The weights were assigned to the edge using the followings.

- Nodes with in-weights larger than 1 (generated by the LFR method) were normalized to 1.
- Moreover, we applied a multiplication with a factor r_w which was a uniform at random number in the interval $[0.6,1]$.

The threshold values of the nodes were generated uniform at random in the interval $[0.05,0.4]$. Two configurations were made:

- number of nodes is fixed to $n = 120$,
- average degree $avgk = 6,7$,
- maximum degree $maxk = 13,10$,
- mixing parameter $\mu_w = 0.1$,
- minimal community size $minc = 7,5$,
- maximal community size $maxc = 21,42$.

5.3 Benchmarking results

The testing of Algorithm 1 using (7) and Algorithm 2 is shown by not only comparing the execution times of these two versions but also the results obtained by two other methods. Namely, the greedy algorithm (Kempe et al., 2003) was implemented, in particular to investigate the lack of submodulatity. The other simple method was a random choice of seed nodes set in 20 times and then the best solution was reported. In all experiments the number of seed nodes were fixed to $k = 2$.

Constraint (9) needs to set up the constant $\varepsilon > 0$. Practically, this should be fixed to slightly bigger than the constraint tolerance value for the solver in use. Since in Gurobi 9.5 the default value for both the feasibility of primal constraints and feasibility of dual constraints is `1e-6` we chose $\varepsilon =$`1e-5` in our experiments.

General observations. We have done some experiments on different formalism and found the following results.

- As it was remarked after the proof of Proposition 1 constraint (8) can be replaced by (3) together with adding for each node i a loop edge with weight equal to θ_i. This version was about 18% longer than using Algorithm 2 as proposed.

Table 1 Benchmarking results for the small-world Watts-Strogatz graphs; optimum values

			random		greedy		Alg. 1+ (7)		Algorithm 2	
s	β	$i.$	σ	$\mathscr{T}$	σ	$\mathscr{T}$	σ	$\mathscr{T}$	σ	$\mathscr{T}$
4	0.1	1	58	15	60	14	58	8	60	14
		2	60	14	60	9	60	9	60	9
		3	2	1	59	15	59	15	59	15
		4	59	15	60	11	59	9	60	11
		5	4	2	60	10	58	8	60	10
4	0.3	1	60	13	60	7	60	7	60	7
		2	3	2	59	12	58	9	59	12
		3	3	2	58	8	58	8	58	8
		4	6	3	58	13	58	9	58	9
		5	58	11	58	9	58	9	59	11
8	0.1	1	2	1	58	10	60	10	60	10
		2	2	1	11	4	60	11	60	11
		3	3	2	60	10	60	10	60	10
		4	3	2	60	9	60	9	60	9
		5	3	2	33	6	60	10	60	10
8	0.3	1	5	4	48	7	60	8	60	8
		2	2	1	60	7	60	7	60	7
		3	2	1	12	5	60	11	60	11
		4	2	1	7	3	60	12	60	12
		5	2	1	6	2	60	9	60	9
12	0.1	1	2	1	4	2	60	9	60	9
		2	2	1	3	2	60	10	60	10
		3	3	2	4	2	60	11	60	11
		4	2	1	2	1	60	12	60	12
		5	2	1	5	2	7	3	7	3
12	0.3	1	3	2	4	2	7	4	7	4
		2	2	1	4	2	8	5	8	5
		3	2	1	4	2	60	10	60	10
		4	2	1	8	3	60	9	60	9
		5	2	1	8	4	60	10	60	10

- We also investigated the idea of replacing (8) and (9) by

$$\sum_{i=1}^{n} x_{i,t-1} + 1 \leq \sum_{i=1}^{n} x_{i,t} \quad (\forall t \in T).$$

  This formalism, on average, resulted in about two times slower running time.

WS graphs. The results obtained for the Watts-Strogatz test graphs are reported in Table 1 and 2.

The random algorithm were able to find the optimal σ^* value in 6.6% of the cases. However, it always missed the minimal diffusion time $\mathscr{T}^*$. The greedy algorithm found the globally optimal (σ^*, $\mathscr{T}^*$) pairs in 33% of the cases. Note the effect of the fact that the submodularity (see Section 1) does not hold for the greedy algorithm due to the DLT diffusion model. Algorithm 1 from Keskin and Güler (2018) using

Table 2 Benchmarking results for the small-world Watts-Strogatz graphs; running times in seconds

s	β	i.	random	greedy	Alg. 1+ (7)	Algorithm 2
4	0.1	1	0.28	4.55	8.8	16.5
		2	0.23	2.66	25.9	32.9
		3	0.01	7.55	105.4	1,271.6
		4	0.22	5.18	15.2	15.8
		5	0.01	3.17	22.9	21.1
4	0.3	1	0.19	2.22	8.3	9.3
		2	0.01	5.24	17.3	407.5
		3	0.02	4.75	6.9	220.6
		4	0.02	6.25	34.6	635.0
		5	0.18	5.70	12.2	70.3
8	0.1	1	0.01	4.06	212.3	704.4
		2	0.01	0.52	1,311.8	1,849.9
		3	0.02	3.83	2,354.0	1,391.3
		4	0.02	3.11	130.2	149.8
		5	0.02	1.25	480.3	1,651.4
8	0.3	1	0.04	1.91	112.8	87.6
		2	0.01	1.96	38.7	30.0
		3	0.01	0.78	1,233.4	5,569.3
		4	0.01	0.30	6,878.8	13,981.4
		5	0.01	0.12	196.6	212.9
12	0.1	1	0.01	0.12	650.1	1,026.1
		2	0.01	0.12	2,066.8	9,343.7
		3	0.02	0.13	5,346.7	22,485.2
		4	0.01	0.03	23,832.7	103,358.9
		5	0.01	0.13	13.2	32.2
12	0.3	1	0.02	0.13	39.3	43.4
		2	0.01	0.13	85.8	45.3
		3	0.01	0.13	3,549.9	4,640.9
		4	0.01	0.30	712.4	871.3
		5	0.01	0.52	974.5	8,511.0

constraint (7) missed the globally optimal solution in 5 cases (meaning 83.3% success rate).

Regarding the running time, see Table 2, obviously the random and the greedy algorithm were really fast. Comparing Algorithm 1 and 2 it can be seen that the corrected version resulted in usually much longer running time. Our Algorithm 2 can be up to 31 times slower. Closer inspection into the results reveal that, for example, for the case $s = 4, \beta = 0.1, i = 3$ our proposed algorithm needed $1,271$ seconds to prove that there is no better solution than $(59, 15)$. For $t > 15$ values the σ value got decreasing. Note that there are cases where the optimal seed set can make the entire graph influenced, i.e., where $\sigma^* = 60$, yet, our proposed algorithm is much slower. For example, in the case $s = 12, \beta = 0.3, i = 5$ it turns out that our algorithm was struggling in the very last iteration - this is certainly caused by the constraint (8). On the other hand, there are five problem instances where our Algorithm 2 was faster.

LFR graphs. The results obtained for the LFR graphs are shown in Table 3 and 4.

The random and greedy algorithms were able to find the optimal σ^* value in 20% and 80% of the cases, respectively. However, the random selection of seed

Table 3 Benchmarking results for the LFR graphs; optimal values

						random		greedy		Alg. 1+ (7)		Algorithm 2	
avgk	*maxk*	μ_w	*minc*	*maxc*	*i.*	σ	$\mathscr{T}$	σ	$\mathscr{T}$	σ	$\mathscr{T}$	σ	$\mathscr{T}$
6	10	0.1	5	42	1	110	13	120	13	110	9	120	13
					2	103	12	103	10	103	10	103	10
					3	120	12	120	9	120	8	120	8
					4	19	6	91	12	90	10	91	12
					5	49	15	101	11	101	11	101	11
7	13	0.1	7	21	1	75	10	94	11	96	10	96	10
					2	62	13	120	13	120	13	120	13
					3	23	6	120	15	120	15	120	15
					4	2	1	83	8	83	8	83	8
					5	51	9	111	10	111	10	120	18

Table 4 Benchmarking results for the LFR graphs; runnig times in seconds

avgk	*maxk*	μ_w	*minc*	*maxc*	*i.*	random	greedy	Alg. 1+ (7)	Algorithm 2
6	10	0.1	5	42	1	0.62	19.77	330.4	413.6
					2	0.49	20.05	785.6	5,099.7
					3	0.42	11.39	190.7	193.9
					4	0.19	11.37	135.8	398.8
					5	0.68	30.39	1,886.5	18,279.5
7	13	0.1	7	21	1	0.36	10.19	200.6	1,214.2
					2	0.59	23.74	320.0	388.9
					3	0.21	23.79	451.4	579.6
					4	0.02	15.16	108.2	979.8
					5	0.33	29.67	219.5	420.4

nodes always missed the corresponding minimal diffusion time $\mathscr{T}^*$. The greedy algorithm found the globally optimal (σ^*, $\mathscr{T}^*$) pairs in 70% of the cases. Note that greedy reported larger diffusion time than the optimal in two cases. We can see that Algorithm 1 using (7) missed the globally optimal solution for 3 graphs.

Regarding the running times, see Table 4, random and greedy were again really fast. Comparing Algorithm 1 and 2 we can see that our proposed version can be up to 10 times slower. This is due to the same fact mentioned for the WS graphs as well: it takes considerable time to prove the optimality of the found solution.

5.4 Possibility for improvement?

The exact ILP model can provide us with rather pessimistic running times even for the relatively small graphs used in our benchmarking experiments. It is tempting to suggest that the exact model used in our Algorithm 2 could be combined with either the random selection of the seed nodes or with the greedy approach as finding initial values and hence potentially reduce the overall running time. It is easy to see that this could lead to suboptimal results and the argument is as follows. Recall that the IM problem considered in this paper aims at finding both the optimal value of the maximum influence together with the minimum diffusion time. According to our experiments both random and greedy approach can sometimes overshoot for the diffusion time, i.e., can find a solution for which the diffusion time is larger than the optimal.

6 Conclusions

We proposed an exact 0-1 linear programming model for the influence maximization problem based on deterministic linear threshold model. By rigorous analysis the correctness was shown. The work was inspired by a recent paper (Keskin & Güler, 2018). In fact, our proposed model is an improved version in a sense that the model in Keskin and Güler (2018) does not always find the global optimum. We demonstrated this fact in our analysis and by numerical testings.

According to our benchmarking results, even for relatively small graphs, finding the exact solution can only be done in very pessimistic running times. In one hand this is not surprising as the problem is strongly NP-hard. On the other hand, our exact model is the first one to computationally demonstrate how difficult is to find the global solution.

Acknowledgements. We thank our reviewers for their critical comments and valuable suggestions that highlighted important details and helped us in improving our paper.

The research has been supported by the grant TKP2021-NVA-09 of the Ministry for Innovation and Technology, Hungary and by the grant SNN-135643 of the National Research, Development and Innovation Office, Hungary.

Declaration

Conflict of interest. The authors declare that they have no conflict of interest.

References

Acemoglu, D., Ozdaglar, A., Yildiz, E. (2011). Diffusion of innovations in social networks. *50th IEEE Conference on Decision and Control and European Control Conference* (pp. 2329–2334).

Altarelli, F., Braunstein, A., Dall'Asta, L., Zecchina, R. (2013). Optimizing spread dynamics on graphs by message passing. *Journal of Statistical Mechanics: Theory and Experiment*, P09011.

Chen, C.-L., Pasiliao, E.L., Boginski, V. (2020). A cutting plane method for least cost influence maximization. S. Chellappan, K.-K.R. Choo, & N. Phan (Eds.), *Computational data and social networks* (pp. 499–511). Springer International Publishing.

Cheng, C.-H., Kuo, Y.-H., Zhou, Z. (2020). Outbreak minimization vs influence maximization: an optimization framework. *BMC Medical Informatics and Decision Making*, *20*(1), 1–13.

Farzaneh, G.-B., Masoud, A., Heshaam, F. (2021). MLPR: Efficient influence maximization in linear threshold propagation model using linear programming. *Social Network Analysis and Mining*, *11*.

Fourer, R., Gay, D., Kernighan, B. (1993). *AMPL. a modeling language for mathematical programming*. Thomson.

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*, *12*(3), 211–223.

Granovetter, M. (1978). Threshold models of collective behavior. *American Journal of Sociology*, *83*(6), 1420–1443.

Güney, E. (2019). An efficient linear programming based method for the influence maximization problem in social networks. *Information Sciences*, *503*, 589–605.

Güney, E., Leitner, M., Ruthmair, M., Sinnl, M. (2021). Large-scale influence maximization via maximal covering location. *European Journal of Operational Research*, *289*(1), 144-164.

Gursoy, F., & Gunnec, D. (2018). Influence maximization in social networks under deterministic linear threshold model. *Knowledge-Based Systems*, *161*, 111–123.

Hansen, P., Jaumard, B., Savard, G. (1992). New branch-and-bound rules for linear bilevel programming. *SIAM Journal on Scientific and Statistical Computing*, *13*(5), 1194-1217.

Kahr, M., Leitner, M., Ruthmair, M., Sinnl, M. (2021). Benders decomposition for competitive influence maximization in (social) networks. *Omega*, *100*, 102264.

Karampourniotis, P., Szymanski, B., Korniss, G. (2019). Influence maximization for fixed heterogeneous thresholds. *Scientific Reports*, *9*, 5573.

Kempe, D., Kleinberg, J., Tardos, E. (2003). Maximizing the spread of influence through a social network. *Proceedings of the ninth acm sigkdd international conference on knowledge discovery and data mining* (pp. 137–146).

Keskin, M., & Güler, M. (2018). Influence maximization in social networks: an integer programming approach. *Turkish Journal of Electrical Engineering & Computer Sciences*, *26*(6), 3383–3396.

Lancichinetti, A., Fortunato, S., Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms. *Phys. Rev. E*, *78*, 046110.

Li, Y., Fan, J., Wang, Y., Tan, K.-L. (2018). Influence maximization on social graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, *30*(10), 1852-1872.

Liu, B., Cong, G., Xu, D., Zeng, Y. (2012). Time constrained influence maximization in social networks. *2012 IEEE 12th international conference on data mining* (pp. 439–448).

Lu, Z., Zhang, W., Wu, W., Fu, B., Du, D. (2011, June). Approximation and inapproximation for the influence maximization problem in social networks under deterministic linear threshold model. *2011 31st international conference on distributed computing systems workshops* (pp. 160–165).

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

Michael, K., Markus, L., Ivana, L. (2022). The impact of passive social media users in (competitive) influence maximization. *Technical report. University of Vienna, Austria.*

Nannicini, G., Sartor, G., Traversi, E., Calvo, R. (2020). An exact algorithm for robust influence maximization. *Mathematical Programming*, *183*, 419–453.

Nemhauser, G., Wolsey, L., Fisher, M. (1978). An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, *14*, 265–294.

Qiang, Z., Pasiliao, E.L., Zheng, Q.P. (2019). Model-based learning of information diffusion in social media networks. *Applied Network Science*, *4*(1), 1–16.

Riquelme, F., Gonzalez-Cantergiani, P., Molinero, X., Serna, M. (2018). Centrality measure in social networks based on linear threshold model. *Knowledge-Based Systems*, *140*, 92-102.

Rosa, D., & Giua, A. (2013). On the spread of innovation in social networks. *IFAC Proceedings Volumes*, *46*(27), 322–327.

Shunyu, Y., Neng, F., Jie, H. (2022). Modeling the spread of infectious diseases through influence maximization. *Optimization Letters*, *16*, 1563–1586.

Talukder, A., Alam, M.G.R., Tran, N.H., Niyato, D., Park, G.H., Hong, C.S. (2019). Threshold estimation models for linear threshold-based influential user mining in social networks. *IEEE Access*, *7*, 105441-105461.

Watts, D., & Strogatz, S. (1998). Collective dynamics of small-world-networks. *Nature*, *393*(6684), 440–442.

Wu, H.-H., & Küçükyavuz, S. (2018). A two-stage stochastic programming approach for influence maximization in social networks. *Computational Optimization and Applications*, *69*(3), 563–595.

Xu, R. (2013). An L_p norm relaxation approach to positive influence maximization in social network under the deterministic linear threshold model. A. Bonato, M. Mitzenmacher, & P. Prałat (Eds.), *Algorithms and models for the web graph* (pp. 144–155). Springer International Publishing.

Yang, L., Giua, A., Li, Z. (2017). Minimizing the influence propagation in social networks for linear threshold models. *IFACPapersOnLine*, *50*, 14465–14470.