

## An exact algorithm for robust influence maximization

**Giacomo Nannicini · Giorgio Sartor ·  
Emiliano Traversi · Roberto Wolfler  
Calvo**

Received: date / Accepted: date

**Abstract** We propose a Branch-and-Cut algorithm for the robust influence maximization problem. The influence maximization problem aims to identify, in a social network, a set of given cardinality comprising actors that are able to influence the maximum number of other actors. We assume that the social network is given in the form of a graph with node thresholds to indicate the resistance of an actor to influence, and arc weights to represent the strength of the influence between two actors. In the robust version of the problem that we study, the node thresholds and arc weights are affected by uncertainty and we optimize over a worst-case scenario within given robustness budgets. We study properties of the robust solution and show that even computing the worst-case scenario for given robustness budgets is NP-hard. We implement an exact Branch-and-Cut as well as a heuristic Branch-Cut-and-Price. Numerical experiments show that we are able to solve to optimality instances of size comparable to other exact approaches in the literature for the non-robust problem, and we can tackle the robust version with similar performance. On larger

---

G. Nannicini  
IBM T.J. Watson, Yorktown Heights, NY  
E-mail: nannicini@us.ibm.com

G. Sartor  
SINTEF Digital,  
Oslo, Norway,  
E-mail: giorgio.sartor@sintef.no

E. Traversi  
Laboratoire d'Informatique de Paris Nord,  
Université de Paris 13; and Sorbonne Paris Cité,  
CNRS (UMR 7538), 93430 Villetaneuse, France  
E-mail: traversi@lipn.fr

R. Wolfler Calvo  
Laboratoire d'Informatique de Paris Nord,  
Université de Paris 13; and Sorbonne Paris Cité,  
CNRS (UMR 7538), 93430 Villetaneuse, France  
E-mail: wolfler@lipn.fr

instances ( $\geq 2000$  nodes), our heuristic Branch-Cut-and-Price significantly outperforms a 2-opt heuristic. An extended abstract of this paper appeared in the proceedings of IPCO 2019.

**Keywords** Influence Maximization · Integer Programming · Robust Optimization.

## 1 Introduction

Social networks are an integral part of social analysis, because they play an important role in the spread of, e.g., information, innovation, or purchase decisions. A social network is defined as a graph with actors (or groups of actors) corresponding to nodes, and arcs corresponding to interactions between actors. Interactions may represent different concepts such as friendship, mentor-apprentice, one- or two-way communication, and so on. Recent years have witnessed growing interest in the definition and study of mathematical models to represent the propagation of *influence* – broadly defined – in a social network, as well as in the identification of the actors that can play an important role in facilitating such propagation. This paper concerns the identification of such actors.

The influence maximization problem is defined on a graph with an associated *diffusion process* that models the spread of influence on the graph. A node is defined as *activated* if it is affected by the diffusion process. A subset of the nodes are selected as *seeds*, and their role is to initialize the diffusion process. Influence propagates in a breadth-first manner starting from the seeds. Several rules can be used to model the activation of a node. A commonly used model associates an *activation threshold* to each node, and a nonnegative weight to each arc representing the strength of the interaction; this paper uses such a model. The condition under which a node is activated by its neighbors is often described by an *activation function*, several types of which are discussed in the literature. The Influence Maximization Problem (IMP) is defined as the problem of identifying a subset of nodes of a given cardinality that maximizes the number of nodes activated at the end of the influence propagation process.

*Literature review.* The idea of identifying the set of nodes that maximizes influence on a network dates back to [8] and [21]. Several variants of the IMP have been presented in the literature; we refer to the recent surveys [2], [16] and [18] for an extensive analysis of these variants.

The majority of the literature models the diffusion of influence on the graph using a *threshold* model or a *cascade* model, see e.g., [15]. In the threshold model, a node becomes active if and only if a function of the weights on arcs incoming from activated neighbors is larger than the node threshold. In the cascade model, a node becomes active if at least one of its neighbors is activated. We further distinguish between *deterministic*, *stochastic* and *robust* models. In deterministic models, the graph parameters (i.e., weights and thresholds) are given and immutable. In the stochastic model, some of them

are random variables and we optimize the expected number of activated nodes. In the robust model, some parameters are uncertain and we optimize the worst case over a given uncertainty set (several comprehensive surveys on robust optimization are available in the literature, see e.g., [3, 4, 6, 19]). The distinction between deterministic, stochastic, and robust models is crucial, because the stochastic version of the problem leads to a monotone submodular maximization problem under reasonable assumptions [20]. Hence, it admits an efficient  $(1 - \frac{1}{e})$ -approximation using a greedy algorithm, see e.g., [15, 20]. The deterministic and robust version are not known to admit such an approximation in general, and they tend to be much harder to solve in practice (but see [7] for an approximation algorithm for a robust version of IMP under some conditions, that still requires submodularity).

The papers [15, 16] study the greedy approach for the threshold and cascade model in the stochastic setting. [23] proposes an exact cutting plane approach for the same class of models, using strong optimality cuts exploiting submodularity. The papers [7, 14] present a greedy algorithm for a robust version of the cascade models, optimizing a measure of regret regarding the set of chosen seeds.

Among the variants of IMP, we mention the Target Set Selection Problem (TSSP) [1], the (Generalized) Least Cost Influence Problem (GLCIP) in [13, 11], and the Technology Diffusion Problem (TDP) in [12, 17]. The TSSP looks for the minimum-cost set of seed nodes that guarantees activation of a given percentage of the total number of nodes. The TSSP and the IMP are in some sense two different formulations for the same problem [1]: in TSSP, the total number of activated nodes is a constraint and the number of seed nodes is the objective function, while for IMP it is the other way around. GLCIP is a generalization of TSSP that allows *incentives* to decrease node activation thresholds paying a cost [11]. Both [1] and [11] use integer programming formulations with an exponential size. In the TDP, a nodes activates if it is adjacent to a connected component of active nodes of size at least equal to its threshold. The goal is to find a seed set whose initial activation would trigger a cascade activating the entire graph. [12] proposes an  $O(rl \log(n))$ -approximation algorithm, where  $r$  is the diameter of the given graph, and  $l$  is the number of distinct thresholds used in the instance. This result is improved in [17] to a  $O(\min\{r, l\} \log(n))$ -approximation algorithm; to the best of our knowledge, exact algorithms for TDP are not discussed in the literature.

*Contributions of this paper.* We present an exact algorithm for the deterministic and robust IMP assuming a linear threshold model. The algorithm that we propose is based on a mixed-integer linear program (MILP) and Branch-and-Cut. The model of uncertainty for the robust IMP is akin to the  $\Gamma$ -robustness of [5]: we assume that the node activation thresholds and the arc weights are allowed to vary within a certain range, but the total amount of deviation from the nominal problem data is limited by two robustness parameters (one for node thresholds, one for arc weights); our goal is to choose seeds so as to optimize the total influence on the graph, assuming the worst-case realization of

the problem data allowed by the given robustness parameters. It is known that the influence maximization problem is NP-hard [16]; we show that computing the total influence under our robust model, given the set of seeds, is NP-hard as well. We study properties of the robust solutions, formalizing the intuition that the price of robustness is higher for changes in the node thresholds than for changes in the arc weights. To the best of our knowledge, this is the first time that an exact algorithm for a robust version of IMP is proposed in the literature. Furthermore, even the non-robust version of the MILP used in this paper is novel. Our algorithm for the robust IMP is not simply the application of the ideas of [5] to the non-robust model: indeed, for reasons that will become apparent after discussing the mathematical model for IMP in more detail, it is not clear how to apply the procedure of [5] to our model. We therefore propose a full Branch-and-Cut algorithm that computes, at each node, a valid bound for the robust problem, and progressively refines such bound.

The MILP that we propose for IMP originates from a bilevel formulation of the problem, where the inner problem (a linear problem with an exponential number of constraints and a provably integer optimum) is dualized, leading to a quadratic problem with binary and linear variables. This formulation is linearized with the use of indicator constraints. The final model contains an exponential number of variables. We show that they can be generated within a column generation framework. The number of variables depends on the density of the graph and the arc weights. To make the problem robust, we use valid dual bounds that are progressively refined using the solution of a sub-MILP.

We test the proposed Branch-and-Cut algorithm on a set of instances comprising social network graphs taken from [11]. We show that our integer programming formulation for the non-robust model is competitive with the exact algorithm of [11] for the related GLCIP problem, and we are able to solve the robust IMP to optimality for instances of similar size ( $\leq 100$  nodes). Furthermore, we implement a heuristic Branch-Cut-and-Price algorithm that works with a subset of the columns (i.e., decision variables). We apply this heuristic to larger graphs (up to 5000 nodes), which are out of reach for our implementation of the exact algorithm because the set of all columns is prohibitively large. We compare the heuristic Branch-Cut-and-Price to a 2-opt heuristic, and show that it consistently finds significantly better solutions within the same amount of time.

The rest of this paper is organized as follows. Section 2 formally describes the robust IMP. Section 3 discusses the computational complexity of the problem and some properties of the robust solution. Section 4 presents a new formulation for the deterministic version of the IMP. Section 5 extends the formulation to the robust case, proposing a Branch-Cut-and-Price algorithm for its solution. Section 6 gives implementation details about the algorithm presented in the previous section and introduces a heuristic version of the Branch-Cut-and-Price. Section 7 provides a numerical evaluation of the proposed methodology and Section 8 concludes the paper.

```

1:  $A_0 \leftarrow \{j \in V : y_j = 1\}$ 
2: for  $k = 1, \dots, n$  do
3:    $A_k \leftarrow A_{k-1}$ 
4:   for every  $j \in V : A_{k-1} \cap \delta^-(j) \neq \emptyset$  do
5:     if  $\sum_{i' \in \delta^-(j) : i' \in A_{k-1}} w_{i'j} \geq t_j$  then  $A_k \leftarrow A_k \cup \{j\}$ 
return  $A_n$ 

```

**Algorithm 1:** Function INFLUENCESPREAD( $y, t, w$ ).

## 2 Problem formulation

To formulate the problem of maximizing the influence on a graph  $G = (V, E)$ , we start by considering the problem of computing the amount of influence spread once the activation seeds are given. Assume w.l.o.g. that  $V = \{1, \dots, n\}$ , and denote by  $\delta^-(j)$  and  $\delta^+(j)$  the instar and outstar of node  $j$ . In the rest of this paper, we will use  $y \in \{0, 1\}^n$  as the incidence vector of the seeds. Given seeds  $y$ , a vector of node thresholds  $t$ , and arc weights  $w$ , we define the set of active nodes as the set returned by Alg. 1, and the corresponding influence as its cardinality. Notice that the main influence diffusion loop is repeated  $n$  times, but clearly it can be stopped at iteration  $k < n$  if no node is added to  $A_k$ .

The activation function used in Alg. 1 is known as the *linear threshold* model. If the node activation thresholds  $t$  and arc weights  $w$  are given as input, the model is deterministic. This paper studies a robust counterpart of linear threshold model, in which the activation threshold  $t_j$  of each node  $j$  can deviate by some fraction  $\Delta_N$  from its nominal value, and the total amount of threshold variations is upper bounded by a given number  $B_N$ . Similarly, each arc weight  $w_{ij}$  can deviate by some fraction  $\Delta_A$  from its nominal value, and the total amount of weight variations is upper bounded by  $B_A$ . Because we want to optimize over a worst-case scenario, the activation thresholds can only increase and the arc weights can only decrease. Throughout the rest of the paper, we assume w.l.o.g. that  $\Delta_N \in [0, \infty)$  and  $\Delta_A \in [0, 1]$ . Given a vector of seeds  $\bar{y} \in \{0, 1\}^n$ , the total amount of influence that spreads on the graph under this robust setting is the optimum of the following problem:

$$\left. \begin{array}{l}
\text{RI}_{x,\theta,\varphi}(\bar{y}) := \\
\forall j \in V \quad \sum_{i \in \delta^-(j)} (w_{ij}x_i - \varphi_{ij}) - \theta_j + \epsilon - Mx_j \leq t_j \\
\forall j \in V \quad x_j \geq \bar{y}_j \\
\forall j \in V \quad \sum_{j \in V} \theta_j \leq B_N \\
\forall j \in V \quad 0 \leq \theta_j \leq \Delta_N t_j \\
\forall i, j \in V, i \neq j \quad \sum_{i,j \in V, i \neq j} \varphi_{ij} \leq B_A \\
\forall j \in V \quad 0 \leq \varphi_{ij} \leq \Delta_A w_{ij} x_i \\
\forall j \in V \quad x_j \in \{0, 1\},
\end{array} \right\} \quad (1)$$

where a tight big- $M$  coefficient can be computed as  $\sum_{i \in \delta^-(j)} w_{ij} + \epsilon$ . In the above formulation,  $\epsilon$  is a small enough value ensuring that if  $x_j = 0$  then

$\sum_{i \in \delta^-(j)} w_{ij} x_i < t_j$ . For example, if the node thresholds and arc weights are decimal numbers with  $d$  digits of precision after the decimal dot, one can pick  $\epsilon = 10^{-d}$  because the difference between the two sides of the equation will be at least  $10^{-d}$ . Notice that this effectively imposes a lower bound on the minimum node threshold change. We use  $10^{-d}/2$  in our numerical tests.

For each node  $j$ , we use variable  $\theta_j$  to denote the increase of node threshold  $t_j$ , and for each arc  $(i, j)$ , we use variable  $\varphi_{ij}$  to represent the decrease of weight  $w_{ij}$ . The constraints  $\sum_{j \in V} \theta_j \leq B_N, 0 \leq \theta_j \leq \Delta_N t_j, \forall j \in V$  and  $\sum_{j \in V} \varphi_{ij} \leq B_A, 0 \leq \varphi_{ij} \leq \Delta_A w_{ij} x_i, \forall j \in V$  define a polyhedral uncertainty set in a manner similar to [5]. We need the binary variable  $x_j$  in the upper bound for  $\varphi_{ij}$  because otherwise, due to the first constraint in (1), the activation of a node could be influenced by an arc coming from an inactive neighbor.

**Proposition 1** *The optimum of  $RI_{x,\theta,\varphi}(\bar{y})$  is the total influence spread on  $G$  from seeds  $\bar{y}$  under our robustness model, and the variables  $x_j$  indicate which nodes are active at the end of the influence propagation process.*

*Proof* It suffices to show that for a given choice of the threshold change  $\theta_j$  and arc weight change  $\varphi_{ij}$ , problem  $RI_{x,\theta,\varphi}(\bar{y})$  computes the total influence spread as if we had applied  $INFLUENCESPREAD(\bar{y}, t + \theta, w + \varphi)$ ; the result then follows because we are minimizing over  $\theta$  and  $\varphi$ , thereby yielding the influence in the worst-case realization of uncertainty.

Notice that  $RI_{x,\theta,\varphi}(\bar{y})$  is a minimization problem and each  $x_j$  is lower bounded by two quantities only:  $\bar{y}_j$ , and  $\frac{\sum_{i \in \delta^-(j)} (w_{ij} x_i - \varphi_{ij}) - t_j - \theta_j + \epsilon}{\sum_{i \in \delta^-(j)} w_{ij}}$ . The latter quantity is  $> 0$  if and only if  $\sum_{i \in \delta^-(j)} (w_{ij} x_i - \varphi_{ij}) > t_j + \theta_j$ . Notice that, because of the constraints  $\varphi_{ij} \leq \Delta_N x_i$ , this expression is equivalent to  $\sum_{i \in \delta^-(j)} (w_{ij} - \varphi_{ij}) x_i > t_j + \theta_j$ , implying that  $x_j = 1$  if and only if its linear activation rule (with weights  $w + \varphi$  and threshold  $t + \theta$ ) is triggered by its neighbors. If we apply  $INFLUENCESPREAD(\bar{y}, t + \theta, w + \varphi)$ , it is easy to see by induction over the main loop that for each  $x_j \in A_k$  there is an implied lower bound  $x_j \geq 1$  (recall that  $x_j$  is binary), and for all nodes  $\notin A_n$  there is no such implied lower bound. It follows that in the optimal solution  $x_j = 1$  if and only if  $j \in A_n$ .  $\square$

The above formulation does not suffer from self-activating loops (which require additional caution in some other formulations, see e.g., [11]), and it has  $n$  binary variables only. Problem  $RI_{x,\theta,\varphi}(\bar{y})$  is stated as a MILP; Section 3.1 shows that computing its value is NP-hard, hence using a MILP formulation is justified. Because of Prop. 1, the robust IMP with  $q$  activation seeds can be solved as the following bilevel optimization problem:

$$\left. \begin{array}{l} \max_y RI_{x,\theta,\varphi}(y) \\ \forall j \in V \quad \sum_{j \in V} y_j = q \\ \quad \quad \quad y_j \in \{0, 1\}. \end{array} \right\} \quad (\text{R-IMP})$$

Notice that if we fix  $\theta = 0$  and  $\varphi = 0$ , solving  $RI_{x,\theta=0,\varphi=0}(\bar{y})$  yields  $x_j = 1$  exactly for the nodes returned by  $INFLUENCESPREAD(\bar{y}, t, w)$ . It is easy to

show by counterexample that  $\text{RI}_{x,\theta,\varphi}(y)$ , taken as a set function of the incidence vector  $y$ , is not submodular even for fixed  $\theta$  and  $\varphi$ . The robust approach of [5] is difficult to apply here because we do not even have a single-level formulation for the problem. It may be possible to overcome this difficulty with a different formulation with an increased number of variables: in particular, a time-expanded graph with  $n^2$  nodes to represent the  $n$  iterations of Alg. 1 would in principle lead to a single-level formulation, which might allow different strategies to make the problem robust. Our approach keeps a model with  $n$  binary variables: this is the subject of Sections 4 and 5.

### 3 Properties of the robust problem

Before developing an algorithm for (R-IMP), we study some of its properties. More specifically, we show that the inner problem  $\text{RI}_{x,\theta,\varphi}(\bar{y})$  is NP-hard given  $\bar{y}$ . We then turn our attention to properties of the robust solution under different uncertainty sets, formalizing the natural intuition that the price of robustness is higher for node threshold uncertainty than it is for arc weight uncertainty.

#### 3.1 Hardness

We define the decision version of  $\text{RI}_{x,\theta,\varphi}(y)$  as follows:

**Problem (Robust Influence): determine the influence spread under a robust model.**

Input: graph  $G = (V, E)$  with node thresholds  $t$ , arc weights  $w$ , vector of seed nodes  $\bar{y} \in \{0, 1\}^n$ , robustness budget  $B_N, B_A$ , maximum node threshold and arc weight deviation  $\Delta_N, \Delta_A$ , and an integer  $k$ .

Output: is  $\text{RI}_{x,\theta,\varphi}(\bar{y}) \leq k$ ?

**Theorem 1** *The problem “Robust Influence” is NP-complete, even if  $B_N = 0$  or  $B_A = 0$ .*

*Proof* Membership in NP is straightforward: a polynomial-size certificate for the answer can be constructed by providing the optimal  $\theta^{\bar{y}}$  and  $\varphi^{\bar{y}}$ . The answer can be verified by running the polynomial-time algorithm  $\text{INFLUENCESPREAD}(\bar{y}, t + \theta, w + \varphi)$ . To show hardness, we provide a reduction from “Exact Cover by 3-Sets.”

**Problem (Exact Cover by 3-Sets): determine an exact set covering.**

Input: Ground set  $X$  with  $|X| = 3q$ , collection  $\mathcal{C} \subset 2^X$  of subsets of  $X$  with cardinality 3.

Output: does there exist  $C \subset \mathcal{C}$  such that  $\bigcup_{S \in C} S = X$ , and  $|C| = q$ ?

We first examine the case  $B_A = 0$ ; the case  $B_N = 0$  is similar. We construct an instance of Robust Influence as follows; an example is given in Fig. 1. The

graph  $G$  has  $|\mathcal{C}| + 3q + 1$  nodes and has three layers, with arcs always directed toward subsequent layers. The first layer contains only a special node, selected as the only seed node for the influence spread. The special node is connected to  $|\mathcal{C}|$  nodes in the second layer, representing the collection of subsets of  $X$  with an arc with weight 2. The  $|\mathcal{C}|$  nodes are connected to  $|X|$  nodes in the third layer, representing the elements of  $X$ . A node in the second layer, corresponding to  $S \in \mathcal{C}$ , is connected to the nodes representing the three elements of  $X$  that it contains. All the arcs between the second and third layer have weight 1. All node thresholds in the second layer are 1, all node thresholds in the third layer are equal to the respective indegree. We set  $B_N = q(1 + \epsilon)$ ,  $k = |\mathcal{C}| - q + 1$ ,  $\Delta_N = 2$ ,  $\Delta_A = 1$ .

Suppose there exists an exact cover  $C$  of  $X$ . Then we can set  $\theta_j = 1 + \epsilon$  for all  $j \in \mathcal{C}$ ; in other words, we increase by  $1 + \epsilon$  the threshold of all nodes in the graph corresponding to the subsets in  $\mathcal{C}$  that yield an exact cover of  $X$ . By doing so, these nodes cannot be activated. In the third layer, a node is active if and only if all the entering arcs are coming from active nodes. But because  $C$  is an exact cover, there will be exactly one inactive arc for each node in the third layer. Thus, all nodes in the third layer are inactive. It follows that the total influence spread is  $|\mathcal{C}| - q + 1$ : there is one active node in the first layer (the seed), and  $|\mathcal{C}| - q$  in the second layer.

Conversely, suppose the influence spread is  $|\mathcal{C}| - q + 1$ . Notice that the node in the first layer (chosen as the seed) and at least  $|\mathcal{C}| - q$  nodes in the second layer are always active, because the budget  $B_N$  allows us to deactivate at most  $q$  nodes in the second layer. Hence, these must be the only active nodes. But then all nodes in the third layer must be adjacent to an inactive node in the second layer. It follows that we found an exact cover  $C$  of  $X$ .

The proof for  $B_N = 0, B_A > 0$  is similar: we simply set  $B_N = 0, B_A = q(1 + \epsilon)$ . In this case, we can ensure nodes in the second layer are inactive by decreasing the weight of the corresponding entering arc by  $(1 + \epsilon)$ .  $\square$

### 3.2 Sensitivity Analysis

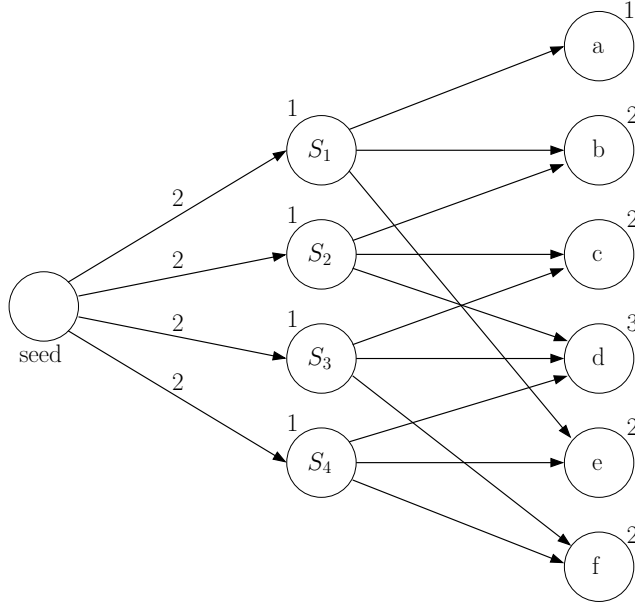
Given a set of robustness parameters  $P = \{B_N, \Delta_N, B_A, \Delta_A\}$ , we denote by  $\text{RI}_{x,\theta,\varphi}^P(y)$  the problem  $\text{RI}_{x,\theta,\varphi}(y)$  with those parameters. In this section we study the impact of  $P$  on the optimal value.

We start by noting that if the upper bounds on the relative deviations  $\Delta_N, \Delta_A$  are nonrestrictive, the price of robustness for a single unit of robustness budget on the node thresholds is larger than on arc weights.

**Proposition 2** *Let  $P^1 = \{B_N^1, \Delta_N^1 = +\infty, B_A^1, \Delta_A^1 = 1\}$ ,  $P^2 = \{B_N^2, \Delta_N^2 = +\infty, B_A^2, \Delta_A^2 = 1\}$ . We have the following implication:*

$$\begin{cases} B_N^1 + B_A^1 = B_N^2 + B_A^2 \\ B_N^1 \leq B_N^2 \end{cases} \implies \text{RI}_{x,\theta,\varphi}^{P^1}(y) \geq \text{RI}_{x,\theta,\varphi}^{P^2}(y).$$





**Fig. 1** Example of the reduction from “Exact Cover by 3-Sets.” The ground set is  $X = \{a, b, c, d, e, f\}$ ,  $q = 2$ ,  $\mathcal{C} = \{S_1, S_2, S_3, S_4\}$  with  $S_1 = \{a, b, e\}$ ,  $S_2 = \{b, c, d\}$ ,  $S_3 = \{c, d, f\}$ ,  $S_4 = \{d, e, f\}$ . Node thresholds are indicated next to each node, and arc weights that are not shown in the figure are 1.

*Proof* We show that for any feasible solution of  $\text{RI}_{x,\theta,\varphi}^{P_1}(y)$ , we can construct a feasible solution for  $\text{RI}_{x,\theta,\varphi}^{P_2}(y)$  with the same objective function value. This immediately implies the claim. Let  $(x^1, \theta^1, \varphi^1)$  be feasible for  $\text{RI}_{x,\theta,\varphi}^{P_1}(y)$ . We construct  $(x^2, \theta^2, \varphi^2)$  feasible for  $\text{RI}_{x,\theta,\varphi}^{P_2}(y)$  as follows:

$$x^2 = x^1 \quad ; \quad \varphi^2 = \frac{B_A^2}{B_A^1} \varphi^1 \quad ; \quad \theta_j^2 = \theta_j^1 + \frac{B_A^1 - B_A^2}{B_A^1} \sum_{i \in \delta^-(j)} \varphi_{ij}^1, \quad \forall j \in V. \quad (2)$$

It is immediate to check that  $\text{RI}_{x,\theta,\varphi}^{P_1}(y) = \text{RI}_{x,\theta,\varphi}^{P_2}(y)$ , and that the point satisfies the constraints  $0 \leq \theta_j^2$ ,  $0 \leq \varphi_{ij}^2$  and  $\sum_{i \in \delta^-(j)} (w_{ij} x_i^2 - \varphi_{ij}^2) - \theta_j^2 + \epsilon - (\sum_{i \in \delta^-(j)} w_{ij} + \epsilon) x_j^2 \leq t_j$ .  $\square$

To prove Prop. 2 we had to assume  $\Delta_N = \infty$  and  $\Delta_A = 1$ . If these conditions are not satisfied, the argument fails because the point constructed in the proof for  $P^2$  may not satisfy the constraints  $\theta \leq \Delta_N t$  and  $\varphi \leq \Delta_A w x$  in the definition of  $\text{RI}_{x,\theta,\varphi}(y)$ . The next two results give sufficient conditions on  $\Delta_N$  and  $\Delta_A$  to be able to perform similar constructions. While the statements are technical, we will discuss some more intuitive special cases at the end of this section.

**Theorem 2** Let  $P^1 = \{B_N^1, \Delta_N^1, B_A^1, \Delta_A^1\}$ ,  $P^2 = \{B_N^2, \Delta_N^2, B_A^2, \Delta_A^2\}$ . We have the following implication:

$$\left\{ \begin{array}{l} \Delta_N^2 \geq \Delta_N^1 + \frac{B_A^1 - B_A^2}{B_A^1} \Delta_A^1 \max_{j \in V} \left\{ \frac{\sum_{i \in \delta^-(j)} w_{ij}}{t_j} \right\} \\ B_N^1 + B_A^1 = B_N^2 + B_A^2 \\ B_N^1 \leq B_N^2 \end{array} \right. \implies RI_{x,\theta,\varphi}^{P^1}(y) \geq RI_{x,\theta,\varphi}^{P^2}(y).$$

*Proof* As for Prop. 2, we show that for any feasible solution of  $RI_{x,\theta,\varphi}^{P^1}(y)$  we can construct a feasible solution for  $RI_{x,\theta,\varphi}^{P^2}(y)$  with the same objective function value. For a feasible solution  $(x^1, \theta^1, \varphi^1)$ , define  $(x^2, \theta^2, \varphi^2)$  as in equation (2). We have to show that the upper bounding constraints in  $RI_{x,\theta,\varphi}^{P^2}(y)$  hold for the proposed solution  $(x^2, \theta^2, \varphi^2)$ . We have:

$$\sum_{j \in V} \theta_j^2 = \sum_{j \in V} \theta_j^1 + \frac{B_A^1 - B_A^2}{B_A^1} \sum_{j \in V} \sum_{i \in \delta^-(j)} \varphi_{ij}^1 \leq B_N^1 + \frac{B_A^1 - B_A^2}{B_A^1} B_A^1 = B_N^2$$

and

$$\varphi_{ij}^2 = \frac{B_A^2}{B_A^1} \varphi_{ij}^1 \leq \frac{B_A^2}{B_A^1} B_A^1 = B_A^2.$$

Finally, since  $(x^1, \theta^1, \varphi^1)$  is feasible, we have:

$$\frac{\theta_j^2}{t_j} = \frac{\theta_j^1}{t_j} + \frac{B_A^1 - B_A^2}{B_A^1} \sum_{i \in \delta^-(j)} \frac{\varphi_{ij}^1}{t_j} \leq \Delta_N^1 + \frac{B_A^1 - B_A^2}{B_A^1} \sum_{i \in \delta^-(j)} \frac{\Delta_A^1 w_{ij} x_{ij}^1}{t_j} \leq \Delta_N^2.$$

Thus,  $\theta^2$  satisfies the constraints  $\theta_j^2 \leq \Delta_N^2 t_j$ .  $\square$

**Theorem 3** Let  $P^1 = \{B_N^1, \Delta_N^1, B_A^1, \Delta_A^1\}$ ,  $P^2 = \{B_N^2, \Delta_N^2, B_A^2, \Delta_A^2\}$ . We have the following implication:

$$\left\{ \begin{array}{l} \Delta_N^2 \geq \Delta_N^1 + \frac{B_A^1 - B_A^2}{B_A^1} \frac{1}{1 - \Delta_A^1} \\ B_N^1 + B_A^1 = B_N^2 + B_A^2 \\ B_N^1 \leq B_N^2 \end{array} \right. \implies RI_{x,\theta,\varphi}^{P^1}(y) \geq RI_{x,\theta,\varphi}^{P^2}(y).$$

*Proof* We use the same approach and notation as in the proof of Thm. (2). Construct the following solution  $(x^2, \theta^2, \varphi^2)$  for  $RI_{x,\theta,\varphi}^{P^2}(y)$ :

$$x^2 = x^1 ; \varphi^2 = \frac{B_A^2}{B_A^1} \varphi^1 ; \theta_j^2 = \begin{cases} \theta_j^1 + \frac{B_A^1 - B_A^2}{B_A^1} \sum_{i \in \delta^-(j)} \varphi_{ij}^1 & \text{if } x_j^1 = 1 \\ 0 & \text{if } x_j^1 = 0 \end{cases}, \forall j \in V.$$

It is immediate to check that  $RI_{x,\theta,\varphi}^{P^1}(y) = RI_{x,\theta,\varphi}^{P^2}(y)$ , and that all the constraints of  $RI_{x,\theta,\varphi}^{P^2}(y)$  except for  $\theta_j^2 \leq \Delta_N^2 t_j$  are respected. Since  $(x^1, \theta^1, \varphi^1)$  is feasible, we have  $\varphi^1 \leq \Delta_A^1 w x^1 \leq \Delta_A^1 w$ , which implies:

$$(1 - \Delta_A^1) \sum_{i \in \delta^-(j)} w_{ij} \leq \sum_{i \in \delta^-(j)} (w_{ij} - \varphi_{ij}^1).$$

Notice that if node  $j$  is not active (i.e.,  $x_j^2 = x_j^1 = 0$ ), we must have  $\sum_{i \in \delta^-(j)} (w_{ij} - \varphi_{ij}^1) < t_j$ , and therefore:

$$\frac{t_j}{(1 - \Delta_A^1)} \geq \sum_{i \in \delta^-(j)} w_{ij} \geq \sum_{i \in \delta^-(j)} \varphi_{ij}^1,$$

where we used the fact that  $\varphi_{ij}^1 \leq \Delta_A^1 w_{ij} x_i^1 \leq \Delta_A^1 w_{ij}$  and  $0 \leq \Delta_A \leq 1$ . Finally, this implies:

$$\frac{\theta_j^2}{t_j} = \frac{\theta_j^1}{t_j} + \frac{B_A^1 - B_A^2}{B_A^1} \sum_{i \in \delta^-(j)} \frac{\varphi_{ij}^1}{t_j} \leq \Delta_N^1 + \frac{B_A^1 - B_A^2}{B_A^1} \sum_{i \in \delta^-(j)} \frac{\varphi_{ij}^1}{t_j} \leq \Delta_N^2. \square$$

The sufficient conditions given in Theorems 2 and 3 can be unintuitive to grasp. We now give two special cases (one for each of the two sufficient conditions) that consider either  $B_N = 0$  or  $B_A = 0$ . These special cases are easier to parse and still capture the essence of the results: as long as the maximum deviation parameters satisfy certain conditions, then the solution to the robust problem, with a given node threshold robustness budget, will be worse (i.e., more conservative) than a solution obtained with the same robustness budget assigned to on arc weight robustness.

**Corollary 1** Let  $P^A = \{0, 0, B_A, \Delta_A\}$ ,  $P^N = \{B_N, \Delta_N, 0, 0\}$ . We have the following implication:

$$\begin{cases} \Delta_N \geq \Delta_A \max_{j \in V} \left\{ \frac{\sum_{i \in \delta^-(j)} w_{ij}}{t_j} \right\} \\ B_N \geq B_A \end{cases} \implies RI_{x, \theta, \varphi}^{P^A}(y) \geq RI_{x, \theta, \varphi}^{P^N}(y).$$

**Corollary 2** Let  $P^A = \{0, 0, B_A, \Delta_A\}$ ,  $P^N = \{B_N, \Delta_N, 0, 0\}$ . We have the following implication:

$$\begin{cases} \Delta_N \geq \frac{1}{1 - \Delta_A} \\ B_N \geq B_A \end{cases} \implies RI_{x, \theta, \varphi}^{P^A}(y) \geq RI_{x, \theta, \varphi}^{P^N}(y).$$

#### 4 Activation set formulation for the non-robust problem

Our first step toward solving (R-IMP) is a formulation for the non-robust counterpart of the problem. In this section we therefore assume that  $\theta, \varphi = 0$  for simplicity. The node activation threshold at node  $j$  is then  $t_j$  and the arc weights are given by the vector  $w$ . For all  $j \in V$ , let  $\mathcal{C}_j : \{S \subseteq \delta^-(j) : \sum_{i \in S} w_{ij} \geq t_j, S \text{ is minimal}\}$ . It is obvious that a node  $j$  is active if and only if there exists  $S \in \mathcal{C}_j$  such that all nodes in  $S$  are active. This is called a *minimal activation set*. The concept of activation set was first introduced in the recent paper [11]. We developed the idea independently and we use our simpler definition, but it is easy to verify that the definition above corresponds to the minimal influencing set of [11] in the context of the linear threshold model

and no incentives. We can reformulate  $\text{RI}_{x,\theta=0,\varphi=0}(\bar{y})$  (see equation (1)) using minimal activation sets.

$$\left. \begin{aligned} \text{AS}_x(\bar{y}) &:= \min \sum_{j \in V} x_j \\ \forall j \in V, \forall S \in \mathcal{C}_j & \sum_{i \in S} x_i - x_j \leq |S| - 1 \\ \forall j \in V & x_j \geq \bar{y}_j. \end{aligned} \right\} \quad (3)$$

**Proposition 3** *If  $\bar{y}$  is a 0-1 vector, the optimal solution  $x^*$  to  $\text{AS}_x(\bar{y})$  is integer and  $x_j = 1$  if and only if  $j \in A_n$  as returned by  $\text{INFLUENCESPREAD}(\bar{y}, t, w)$ .*

*Proof* Every  $x_j$  is lower bounded by  $\bar{y}_j$  and by  $\sum_{i \in S} x_i - |S| + 1$  for some subset of nodes  $S$  adjacent to node  $j$ .

We first show by induction for  $k = 1, \dots, n$  that for every node  $j \in A_k$  in  $\text{INFLUENCESPREAD}(\bar{y}, t, w)$ , we have an implied lower bound  $x_j \geq 1$  in  $\text{AS}_x(\bar{y})$ .

For  $k = 1$  the claim is obvious because of the constraints  $x_j \geq \bar{y}_j$ . To go from  $k-1$  to  $k$ , notice that if node  $j$  is added to  $A_k$  at step  $k$  of  $\text{INFLUENCESPREAD}(\bar{y}, t, w)$ , it must be that  $\sum_{i \in \delta^-(j): i \in A_{k-1}} w_{ij} \geq t_j$ . By the induction hypothesis for all  $i \in A_{k-1}$  we have  $x_i \geq 1$ , hence  $\sum_{i \in \delta^-(j)} w_{ij} x_i \geq t_j$ . By definition of minimal activation set, there must exist some  $S \in \mathcal{C}_j$ , say  $\bar{S}$ , such that  $\bar{S} \subseteq A_{k-1}$ . Then the corresponding constraint  $\sum_{i \in \bar{S}} x_i - x_j \leq |\bar{S}| - 1$  in the formulation  $\text{AS}_x(\bar{y})$  reads  $|\bar{S}| - x_j \leq |\bar{S}| - 1$ , implying  $x_j \geq 1$ .

Finally, for every  $j \notin A_n$ , all the constraints  $\sum_{i \in S} x_i - x_j \leq |S| - 1$  are slack because there does not exist  $S \in \mathcal{C}_j, S \subseteq A_k$  for some  $k$ . Hence, the implied lower bound for  $x_j$  is 0. Since we are minimizing  $\sum_{j \in V} x_j$ , at the optimum  $x_j = 1$  if and only if  $j \in A_n$ .  $\square$

We can use  $\text{AS}_x(\bar{y})$  to obtain a single-level linear formulation for the restriction of (OPT) in which  $\theta = 0$  and  $\varphi = 0$ . More specifically, we consider the following problem:

$$\left. \begin{aligned} \max_y & \text{AS}_x(y) \\ \forall j \in V & \sum_{j \in V} y_j = q \\ & y_j \in \{0, 1\}. \end{aligned} \right\} \quad (\text{IMP-}\theta\text{0-}\varphi\text{0})$$

We first take the dual of the inner problem  $\text{AS}_x(\bar{y})$  for a fixed  $\bar{y}$ . The dual is:

$$\left. \begin{aligned} \max_{\pi, \mu} & \sum_{j \in V} \sum_{S \in \mathcal{C}_j} (|S| - 1) \pi_{j,S} + \sum_{j \in V} \mu_j \bar{y}_j \\ \forall j \in V & \sum_{k \in \delta^+(j)} \sum_{S \in \mathcal{C}_k: j \in S} \pi_{k,S} - \sum_{S \in \mathcal{C}_j} \pi_{j,S} + \mu_j \leq 1 \\ \forall j \in V, \forall S \in \mathcal{C}_j & \pi_{j,S} \leq 0 \\ \forall j \in V & \mu_j \geq 0. \end{aligned} \right\}$$

The solution of this problem has value equal to that of its primal problem and therefore, by Prop. 3, to  $\text{INFLUENCESPREAD}(\bar{y}, t, w)$  whenever  $\bar{y} \in \{0, 1\}^n$ . It follows that a valid formulation for (IMP- $\theta$ 0- $\varphi$ 0) is the following:

$$\left. \begin{aligned} \max_{\pi, \mu, y} & \sum_{j \in V} \sum_{S \in \mathcal{C}_j} (|S| - 1) \pi_{j,S} + \sum_{j \in V} \mu_j y_j \\ \forall j \in V & \sum_{k \in \delta^+(j)} \sum_{S \in \mathcal{C}_k: j \in S} \pi_{k,S} - \sum_{S \in \mathcal{C}_j} \pi_{j,S} + \mu_j \leq 1 \\ & \sum_{j \in V} y_j = q \\ \forall j \in V, \forall S \in \mathcal{C}_j & \pi_{j,S} \leq 0 \\ & \mu_j \geq 0 \\ & y_j \in \{0, 1\}. \end{aligned} \right\}$$

This is a quadratic problem, but we can easily reformulate it as a linear problem with indicator constraints. To do so, we simply notice that whenever  $y_j = 0$ ,  $\mu_j$  has no contribution in the objective function; since  $\mu_j$  appears in a single constraint and increasing  $\mu_j$  reduces the feasible region for the remaining variables, there exists an optimal solution in which  $y_j = 0$  implies  $\mu_j = 0$ . Thus, we obtain the following formulation for (IMP- $\theta$ - $\varphi$ 0):

$$\left. \begin{array}{l} \max_{\pi, \mu, y} \quad \sum_{j \in V} \sum_{S \in \mathcal{C}_j} (|S| - 1) \pi_{j,S} + \sum_{j \in V} \mu_j \\ \forall j \in V \quad \sum_{k \in \delta^+(j)} \sum_{S \in \mathcal{C}_k: j \in S} \pi_{k,S} - \sum_{S \in \mathcal{C}_j} \pi_{j,S} + \mu_j \leq 1 \\ \forall j \in V \quad y_j = 0 \Rightarrow \mu_j = 0 \\ \sum_{j \in V} y_j = q \\ \forall j \in V, \forall S \in \mathcal{C}_j \quad \pi_{j,S} \leq 0 \\ \forall j \in V \quad \mu_j \geq 0 \\ \forall j \in V \quad y_j \in \{0, 1\}. \end{array} \right\} \text{(DUAL-}\theta\text{-}\varphi\text{0)}$$

The advantage with respect to (R-IMP) is of course that we now have a single-level problem, rather than bilevel. To achieve this result we had to fix  $\theta = 0$  and  $\varphi = 0$ . This restriction will be lifted in the next section.

We prove an additional result that we could not successfully exploit from an empirical point of view, but may be interesting for future research.

**Proposition 4** *For every  $\bar{y} \in \{0, 1\}^n$ , the polyhedron corresponding to the LP obtained by fixing  $y = \bar{y}$  in (DUAL- $\theta$ - $\varphi$ 0) is an integral polyhedron.*

*Proof* We show that for a given 0-1 vector  $\bar{y}$ , the remaining system in (DUAL- $\theta$ - $\varphi$ 0) is total dual integral. This implies that it defines an integral polyhedron [9].

The discussion in this section shows that the dual of (DUAL- $\theta$ - $\varphi$ 0) for fixed  $y = \bar{y}$  is the problem  $AS_x(\bar{y})$  defined in (3). To show total dual integrality of the desired system, we need to show that for any integer value of the r.h.s. of the first set of constraints in  $AS_x(\bar{y})$ , either  $AS_x(\bar{y})$  is infeasible, or it has an optimal solution that is integer.

Let  $b$  be a given vector of integer r.h.s. values for the first set of constraints, which are indexed by  $j \in V, S \in \mathcal{C}_j$ . First, notice that if  $b_{j,S} < 0$  for any  $j, S$ , the problem is infeasible; hence, we only need to consider the case  $b \geq 0$ . We show how to construct an integer optimal solution.

Define  $x^0 := \bar{y}$ . Apply the following algorithm: for  $k = 1, \dots, n$ , (i) set  $x_j^k \leftarrow 0 \forall j$ ; (ii) for  $j \in V, S \in \mathcal{C}_j$ , set  $x_j^k \leftarrow \max\{x_j^k, \sum_{i \in S} x_i^{k-1} - b_{j,S}\}$ . It is clear that this defines an integral vector  $x^n$ . We now show that this solution is optimal. Let  $x^*$  be an optimal solution for the problem with r.h.s.  $b$ . We first show by induction that  $x^k \leq x^*$ . For  $k = 0$  this is obvious as  $x \geq \bar{y} = x^0$  is among the constraints. Assume  $x^{k-1} \leq x^*$  and suppose  $x_h^k > x_h^*$  for some  $h$ . Since  $x_h^k$  is initially 0, it must be that for some  $S$ ,  $x_h^k$  is set to  $\sum_{i \in S} x_i^{k-1} - b_{h,S} > x_h^*$  for the first time. But  $\sum_{i \in S} x_i^{k-1} - b_{h,S} \leq \sum_{i \in S} x_i^* - b_{h,S} \leq x_h^*$ , because  $x^*$  satisfies the constraints; this is a contradiction. It follows that  $x^k \leq x^*$  for all  $k = 1, \dots, n$ . It is easy to check that  $x^n$  is feasible by construction, and therefore it must be optimal.  $\square$

The proof of Prop. 4 shows that the LP obtained from (DUAL- $\theta$ - $\varphi$ 0) for integral  $y$  is total dual integral. Hence, variables  $\pi, \mu$  are automatically integer when  $y$  is integer. However, empirically we did not observe any advantage by imposing  $\pi, \mu$  integer in the MILP solver used in our numerical experiments. It is an open question whether this can be exploited in some solution method.

## 5 Branch-Cut-and-Price for robust influence maximization

For the robust version of the problem it is no longer sufficient to consider only the case  $\theta = 0$  and  $\varphi = 0$ . In fact, we would like to optimize over  $\theta$  and  $\varphi$  as in the original formulation (R-IMP), but it is not obvious how to use the dualization trick described in the previous section when  $\theta$  and  $\varphi$  are not fixed. This is because the activation sets may change with  $\theta$  and  $\varphi$ , since  $\theta$  directly affects the node activation thresholds, while  $\varphi$  affects the edges values. To overcome this difficulty, we propose to work with a modification of (DUAL- $\theta$ - $\varphi$ 0) that includes dual  $\pi_{j,S}$  variables for all the activation sets that may be minimal for any of the possible values of  $\theta$  and  $\varphi$ . We choose an objective function that provides a valid upper (dual) bound for any choice of seed nodes  $\bar{y}$ , and that encodes the objective function for a specific choice of  $\theta$  and  $\varphi$ . This objective function is iteratively changed in the course of Branch-and-Cut to tighten the upper bound. At the same time, we keep track of primal solutions so that at termination we have the vector  $\bar{y}$  attaining the maximum possible  $\text{RI}_{x,\theta,\varphi}(\bar{y})$ , i.e., the maximum influence spread when simultaneously minimizing over  $\theta$  and  $\varphi$ .

To implement this idea we must define an appropriate collection of activation sets. Define  $\mathcal{C}_j^e : \{S \subseteq \delta^-(j) : \exists \theta_j \in [0, \Delta_N t_j] \text{ and } \exists \varphi_{ij} \in [0, \Delta_A w_{ij}] \text{ such that } \sum_{i \in S} (w_{ij} - \varphi_{ij}) \geq t_j + \theta_j, S \text{ is minimal}\}$ . We call this an *extended collection of activation sets*. An algorithm to compute  $\mathcal{C}_j^e$  is given in Alg. 2. For a given node  $j$ , the algorithm starts with an empty set. It recursively adds nodes from  $\delta^-(j)$ , starting from the one connected with the arc of maximum weight, and checking whether each new set is an activation set. The algorithm stops when adding new nodes would make the activation set non-minimal for every possible value of the node threshold and of the arc weights. Indeed, minimality is guaranteed by the fact that nodes are added in decreasing order of the corresponding arc weights.

We then define a family of  $2^n$  possible objective functions, each of which defines a problem equivalent to  $\text{RI}_{x,\theta=\bar{\theta},\varphi=\bar{\varphi}}(y)$  for a fixed choice of  $\bar{\theta}$  and  $\bar{\varphi}$ . In practice the optimization considers one objective function at a time, but this can be locally changed at certain nodes of the Branch-and-Bound tree as long as we ensure that pruning decisions based on the dual bound are correct. This will be explained in Section 5.1.

Note that the number of sets in  $\mathcal{C}_j^e$  grows exponentially with the maximum degree of a node in the graph. In the next subsection, we show that our approach based on row generation and replacement is correct under the as-

sumption that all the sets  $\mathcal{C}_j^e$  are known. In practice, to deal with large graphs we propose a column generation approach, discussed in Sections 5.2 and 6.

```

1: if first_to_add >  $|\delta^-(j)|$  then return false
2: any_generated  $\leftarrow$  false
3: if  $\sum_{i \in S} w_{ij} \geq t_j$  then
4:    $\mathcal{C}_j^e \leftarrow \mathcal{C}_j^e \cup S$ 
5:   any_generated  $\leftarrow$  true
6:   if  $\sum_{i \in S} w_{ij} \geq t_j + \min\{B_N, \Delta_N t_j\} + \min\{B_A, \sum_{i \in S} \Delta_A w_{ij}\}$  then return true
   /* If  $S$  is an activation set for any possible threshold of node  $j$ , we do not need to add
   more items to the set */
7: for  $i \leftarrow$  first_to_add to  $|\delta^-(j)|$  do
8:   next_generated  $\leftarrow$  RECGENEXT( $j, S \cup \{i\}, i+1, \mathcal{C}_j^e$ ) /* Check if adding the next largest
   item would make  $S$  a valid activation set */
9:   if next_generated = false then break else any_generated  $\leftarrow$  true
return any_generated

```

**Algorithm 2:** Function RECGENEXT( $j, S, \text{first\_to\_add}, \mathcal{C}_j^e$ ). We assume w.l.o.g. (up to relabeling) that  $\delta^-(j) := \{1, \dots, |\delta^-(j)|\}$ , and  $\delta^-(j)$  is sorted by decreasing value of  $w_{ij}$ . For a given node  $j$ , the first call to the function should be RECGENEXT( $j, S \leftarrow \emptyset, \text{first\_to\_add} \leftarrow 1, \mathcal{C}_j^e \leftarrow \emptyset$ ).

### 5.1 Row generation and replacement

We now discuss the solution of (R-IMP) via Branch-and-Cut assuming that the sets  $\mathcal{C}_j^e$  are known explicitly. For every  $\bar{y} \in \{0, 1\}^n$ , let  $\theta^{\bar{y}}, \varphi^{\bar{y}}$  be the optimum values of  $\theta, \varphi$  for (1), respectively. Let  $\mathcal{C}_j^{\bar{y}} := \{S \subseteq \delta^-(j) : \sum_{i \in S} (w_{ij} - \varphi_{ij}^{\bar{y}}) \geq t_j + \theta_j^{\bar{y}}, S \text{ is minimal}\} \subseteq \mathcal{C}_j^e$  be the set of activation sets that are minimal for a given  $\theta^{\bar{y}}, \varphi^{\bar{y}}$ . We show that (R-IMP) can be solved via the solution of an exponential number of problems of the form (DUAL- $\theta_0$ - $\varphi_0$ ), with different objective functions. Furthermore, this can in principle be solved using just one Branch-and-Bound tree. Consider the following problem, where  $\bar{y} \in \{0, 1\}^n$ .

$$\begin{array}{l}
 \max_{\pi, \mu, y} \quad \left. \begin{array}{l}
 \sum_{j \in V} \sum_{S \in \mathcal{C}_j^{\bar{y}}} (|S| - 1) \pi_{j,S} + z \\
 \sum_{j \in V} \sum_{S \in \mathcal{C}_j^e \setminus \mathcal{C}_j^{\bar{y}}} |S| \pi_{j,S} + \sum_{j \in V} \mu_j - z \geq 0 \\
 \forall j \in V \quad \sum_{k \in \delta^+(j)} \sum_{S \in \mathcal{C}_k^e : j \in S} \pi_{k,S} - \sum_{S \in \mathcal{C}_j^e} \pi_{j,S} + \mu_j \leq 1 \\
 \forall j \in V \quad y_j = 0 \Rightarrow \mu_j = 0 \\
 \sum_{j \in V} y_j = q \\
 \forall j \in V, \forall S \in \mathcal{C}_j^e \quad \pi_{j,S} \leq 0 \\
 \forall j \in V \quad \mu_j \geq 0 \\
 \forall j \in V \quad y_j \in \{0, 1\}.
 \end{array} \right\} \\
 \text{(R-IMP-}\bar{y}\text{)}
 \end{array}$$

**Theorem 4** For any  $\bar{y} \in \{0, 1\}^n$ , the value of (R-IMP- $\bar{y}$ ) at every integer solution  $\tilde{y}$  is an upper bound to  $RI_{x, \theta, \varphi}(\tilde{y})$  (equation (1)), and if  $\tilde{y} = \bar{y}$ , then the value of (R-IMP- $\bar{y}$ ) is exactly  $RI_{x, \theta, \varphi}(\bar{y})$ .

*Proof* Consider the following LP, obtained by fixing  $y = \tilde{y}$  in (R-IMP- $\bar{y}$ ) and eliminating the redundant variable  $z$ :

$$\left. \begin{array}{l} \max_{\pi, \mu} \quad \sum_{j \in V} \sum_{S \in \mathcal{C}_j^{\tilde{y}}} (|S| - 1) \pi_{j,S} + \\ \quad \sum_{j \in V} \sum_{S \in \mathcal{C}_j^e \setminus \mathcal{C}_j^{\tilde{y}}} |S| \pi_{j,S} + \sum_{j \in V} \mu_j \\ \forall j \in V \quad \sum_{k \in \delta^+(j)} \sum_{S \in \mathcal{C}_k^e, j \in S} \pi_{k,S} - \sum_{S \in \mathcal{C}_j^e} \pi_{j,S} + \mu_j \leq 1 \\ \forall j \in V, \tilde{y}_j = 0 \quad \mu_j = 0 \\ \forall j \in V, \forall S \in \mathcal{C}_j^e \quad \pi_{j,S} \leq 0 \\ \forall j \in V \quad \mu_j \geq 0. \end{array} \right\} \quad (4)$$

This problem is feasible as the all-zero solution is feasible. Using the reverse of the transformations discussed in Sec. 4, we can show that the dual of such an LP is equivalent to the following problem:

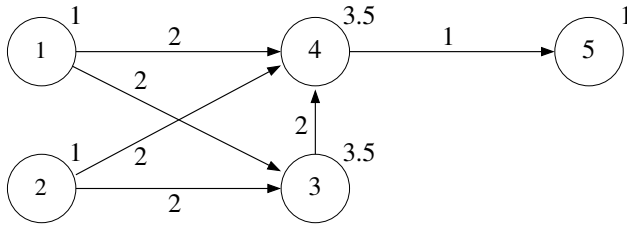
$$\left. \begin{array}{l} \min \quad \sum_{j \in V} x_j \\ \forall j \in V, \forall S \in \mathcal{C}_j^{\tilde{y}} \quad \sum_{i \in S} x_i - x_j \leq |S| - 1 \\ \forall j \in V, \forall S \in \mathcal{C}_j^e \setminus \mathcal{C}_j^{\tilde{y}} \quad \sum_{i \in S} x_i - x_j \leq |S| \\ \forall j \in V \quad x_j \geq y_j^* \\ \forall j \in V \quad x_j \leq 1 \\ \forall j \in V \quad x_j \geq 0. \end{array} \right\} \quad (5)$$

The constraints with r.h.s. value  $|S|$  are redundant and can be dropped. As a result, with the same argument used for Prop. 3, the optimum value of (5) and therefore (4) is equal to  $\text{RI}_{x, \theta = \theta^{\tilde{y}}, \varphi = \varphi^{\tilde{y}}}(\tilde{y})$ . The statement then follows: this value is an upper bound for  $\text{RI}_{x, \theta, \varphi}(\tilde{y})$  (because RI is a minimization problem and  $\theta = \theta^{\tilde{y}}, \varphi = \varphi^{\tilde{y}}$  is just a feasible solution), and is equal to  $\text{RI}_{x, \theta, \varphi}(\tilde{y})$  when evaluated at  $\tilde{y} = \bar{y}$  because the optimal  $\theta, \varphi$  are  $\theta^{\bar{y}}, \varphi^{\bar{y}}$  by definition.  $\square$

Thm. (4) shows a way to obtain valid upper bounds. We can exploit this in the following algorithm for (R-IMP). First, generate the extended collection  $\mathcal{C}_j^e$  of activation sets for all  $j \in V$ , using Alg. 2. It is easy to see that for a given  $\bar{y}$ , each activation set  $S \in \mathcal{C}_j^e$  is valid in the range  $[t_j, \min\{(1 + \Delta_N)t_j, \sum_{i \in S} (w_{ij} - \varphi_{ij}^{\bar{y}})\}]$  and it is minimal if the threshold is strictly greater than  $\sum_{i \in S} (w_{ij} - \varphi_{ij}^{\bar{y}}) - \min_{i \in S} \{w_{ij} - \varphi_{ij}^{\bar{y}}\}$ . Then, implement problem (R-IMP- $\bar{y}$ ) within a Branch-and-Bound solver for an arbitrary initial vector  $\bar{y} \in \{0, 1\}^n$ . At every node of the Branch-and-Bound tree we do the following:

- If the solution to the node LP is not integer, decide if  $\bar{y}$  used in the first row of (R-IMP- $\bar{y}$ ) to define the objective function should be updated, with an arbitrary decision rule.
- If the solution to the node LP is integer, say,  $\tilde{y}$ , then:
  - If  $\bar{y} = \tilde{y}$  for that node, accept the solution;
  - Otherwise, update  $\bar{y} = \tilde{y}$  for that node and its potential children, and put the node back in the list of open nodes. (The solution  $\tilde{y}$  is thus not accepted.)





**Fig. 2** Graph discussed in the example related to Thm. 4. Node labels are indicated inside each node, node thresholds are indicated above and to the right of each node, and arc weights are indicated next to each arc.

Correctness of this algorithm follows by Thm. 4 and standard Branch-and-Bound arguments: we just need to observe that pruning decisions are correct, because an integer solution  $\tilde{y}$  is only accepted if the corresponding LP has value  $\text{RI}_{x,\theta,\varphi}(\tilde{y})$ , and at every node we have a valid upper bound. There are several valid strategies to choose when to replace the objective function that depends on  $\bar{y}$ . For example, it is reasonable to replace  $\bar{y}$  every few levels of the Branch-and-Bound tree with a  $\bar{y}$  (corresponding to a choice of  $\theta, \varphi$ ) that improves the dual bound. This can be computed solving a problem of the form  $\text{RI}_{x,\theta,\varphi}(\hat{y})$  where  $\hat{y}$  is, e.g., the current vector of upper bounds on the  $y$  variables: the cardinality constraint on  $y$  can be temporarily relaxed, since the goal is simply to obtain a choice of  $\theta, \varphi$  that can be used to compute the objective function. In practice, our implementation uses IBM ILOG CPLEX, and CPLEX does not allow changing the objective function in the course of Branch-and-Bound. Thus, we implement a variant of the above algorithm, based on restarts. This is described in Section 6.

As every distinct  $\bar{y}$  yields a different first row in  $(\text{R-IMP-}\bar{y})$ , one may be tempted to keep all these constraints in the formulation so that the objective function  $z$  is upper bounded by the tightest such constraint. However, the following example shows that this would break Thm. 4, in the sense that the linear programming relaxation to the problem would no longer yield valid dual bounds<sup>1</sup>. Consider the graph depicted in Fig. 2. We assume that  $B_N = 0, B_A = 1, q = 2$ . The extended set of activation sets for the three nodes that can be activated by other nodes are:

$$\begin{aligned} \mathcal{C}_3^e &= \{S_1 = \{1, 2\}\} \\ \mathcal{C}_4^e &= \{S_2 = \{1, 2\}, S_3 = \{1, 3\}, S_4 = \{2, 3\}, S_5 = \{1, 2, 3\}\} \\ \mathcal{C}_5^4 &= \{S_6 = \{4\}\} \end{aligned}$$

We consider three different values of  $\bar{y}$  (the other possibilities are irrelevant for the purposes of this example):  $(1, 1, 0, 0, 0), (0, 1, 1, 0, 0), (1, 0, 1, 0, 0)$ . For

<sup>1</sup> In the IPCO version of this paper, all such constraints were added to the formulation as lazy constraints. The discussion here shows that this may yield invalid dual bounds and a potentially suboptimal solution. This is due to a mistake in the proof of Thm. 4 of the IPCO version: it is fixed in this paper. Experiments show that  $\approx 25\%$  of the solutions found in the IPCO paper are suboptimal.

brevity, in the following we will write these solutions as 5-digit binary strings. For each  $\bar{y}$  there are multiple optimal solutions to the problem  $\text{RI}_{x,\theta,\varphi}(\bar{y})$ . In our example, we assume that the optimal solutions have the following values of  $\varphi$  ( $\theta_j$  is always 0 since  $B_N = 0$ ) for each value of  $\bar{y}$ :

$$\begin{aligned}\bar{y} = 11000 &\longmapsto \varphi_{14} = 1 \\ \bar{y} = 01100 &\longmapsto \varphi_{24} = 1 \\ \bar{y} = 10100 &\longmapsto \varphi_{23} = 1.\end{aligned}$$

Notice that the activation sets  $S_2, S_3$  are no longer valid activation sets for node 4 when  $\varphi_{14} = 1$ , as the sum of the corresponding arc weights does not exceed the threshold of node 4. Similarly,  $\varphi_{24} = 1$  disables activation set  $S_4$ , and  $\varphi_{23}$  disables activation set  $S_1$ . Problem  $(\text{R-IMP-}\bar{y})$  with all the constraints corresponding to the three  $\bar{y}$  indicated above reads:

$$\left. \begin{array}{l} \max_{\pi, \mu, y} \\ (\bar{y} = 11000) \quad \pi_{S_1} + 2\pi_{S_2} + 2\pi_{S_3} + \pi_{S_4} + 2\pi_{S_5} + \sum_{j=1, \dots, 5} \mu_j \geq z \\ (\bar{y} = 01100) \quad \pi_{S_1} + \pi_{S_2} + \pi_{S_3} + 2\pi_{S_4} + 2\pi_{S_5} + \sum_{j=1, \dots, 5} \mu_j \geq z \\ (\bar{y} = 10100) \quad 2\pi_{S_1} + \pi_{S_2} + \pi_{S_3} + \pi_{S_4} + 2\pi_{S_5} + \sum_{j=1, \dots, 5} \mu_j \geq z \\ \pi_{S_1} + \pi_{S_2} + \pi_{S_3} + \pi_{S_5} + \mu_1 \leq 1 \\ \pi_{S_1} + \pi_{S_2} + \pi_{S_4} + \pi_{S_5} + \mu_2 \leq 1 \\ \pi_{S_3} + \pi_{S_4} + \pi_{S_5} - \pi_{S_1} + \mu_3 \leq 1 \\ \pi_{S_6} - \pi_{S_2} - \pi_{S_3} - \pi_{S_4} - \pi_{S_5} + \mu_4 \leq 1 \\ -\pi_{S_6} + \mu_5 \leq 1 \\ \forall j = 1, \dots, 5 \quad \sum_{k \in \delta^+(j)} \sum_{S \in \mathcal{C}_k^e: j \in S} \pi_S - \sum_{S \in \mathcal{C}_j^e} \pi_S + \mu_j \leq 1 \\ \forall j = 1, \dots, 5 \quad y_j = 0 \Rightarrow \mu_j = 0 \\ \sum_{j=1, \dots, 5} y_j = 2 \\ \forall j = 1, \dots, 6 \quad \pi_{S_j} \leq 0 \\ \forall j = 1, \dots, 5 \quad \mu_j \geq 0 \\ \forall j = 1, \dots, 5 \quad y_j \in \{0, 1\}. \end{array} \right\} \quad (6)$$

To see where trouble might arise, consider the LP obtained by fixing  $y = 11000$ . We take its dual as it makes the situation easier to understand:

$$\left. \begin{array}{l} \min \\ x_1 + x_2 + x_3 + x_4 + x_5 \\ x_1 + x_2 - x_3 - \gamma_1 - \gamma_2 - 2\gamma_3 \leq 0 \\ x_1 + x_2 - x_4 - 2\gamma_1 - \gamma_2 - \gamma_3 \leq 0 \\ x_1 + x_3 - x_4 - 2\gamma_1 - \gamma_2 - \gamma_3 \leq 0 \\ x_2 + x_3 - x_4 - \gamma_1 - 2\gamma_2 - \gamma_3 \leq 0 \\ x_1 + x_2 + x_3 - x_4 - 2\gamma_1 - 2\gamma_2 - 2\gamma_3 \leq 0 \\ x_4 - x_5 \leq 0 \\ x_1 - \gamma_1 - \gamma_2 - \gamma_3 \geq 0 \\ x_2 - \gamma_2 - \gamma_2 - \gamma_3 \geq 0 \\ \gamma_1 + \gamma_2 + \gamma_3 \geq 1 \\ x, \gamma \geq 0. \end{array} \right\} \quad (7)$$

In the dual (7),  $\gamma$  are the variables corresponding to the first three constraints in (6). The dual has a form similar to (5), as expected, and the constraints correspond to activation sets. However, due to presence of multiple  $\bar{y}$  constraints in (6), the r.h.s. vector of (7) is 0, and the problem is allowed to “choose” a r.h.s. vector using the  $\gamma$  variables. In contrast, the r.h.s. vector of (5) is fixed and corresponds to having a single nonzero  $\gamma$  variable equal to 1. Because of this, the value of (7) can be lower than  $\text{RI}_{x,\theta,\varphi}(\bar{y})$ , yielding an invalid bound, see Thm. 4. In this example, (7) has optimal objective function value 3.5, and the optimal variables are (we report only nonzeros):

$$x_1 = 1, x_2 = 1, x_3 = 0.5, x_4 = 0.5, x_5 = 0.5, \gamma_1 = 0.5, \gamma_3 = 0.5.$$

This is not a valid upper bound, because  $\text{RI}_{x,\theta,\varphi}(11000)$  has value 4, as can be easily checked: only node 3 is inactive at the optimum.

## 5.2 Column generation

As remarked in the preceding discussion,  $\mathcal{C}_j^e$  may have exponentially large cardinality. However, its elements and the related variables can be generated via column generation. We describe the pricing problem for the LP relaxation of (R-IMP- $\bar{y}$ ). Notice that since we have indicator variables in (R-IMP- $\bar{y}$ ), it is not immediately apparent how to obtain the LP relaxation. For all practical purposes, however, indicator variables can be treated as if they were involved in a big-M formulation to activate/deactivate constraints. In the specific case of (R-IMP- $\bar{y}$ ), we can simply assume that if a variable  $y_j$  has value 0 at node  $j$ , then  $\mu_j = 0$ , and otherwise  $\mu_j \geq 0$ . Thus, at any node of the Branch-and-Bound tree, we have access to the corresponding LP relaxation.

Since the variables  $\pi_{j,S}$  are defined for each activation set  $S$  of node  $j$ , we can study column generation by considering a single node  $j \in V$ : the same reasoning applies to all nodes of the graph. To generate  $\pi_{j,S}$ , we need to determine its activation set  $S$ . Call  $x$  the dual variables for the first set of constraints of (R-IMP- $\bar{y}$ ) (i.e., the constraints for  $j \in V$ , rather than the constraint used to bound the objective function variable  $z$ ; we use  $x$  as these constraints correspond to the variables in (3)). The reduced cost of  $\pi_{j,S}$  in  $\mathcal{C}_j^e$  is:

$$x_j - \sum_{i \in S} x_i - \begin{cases} (|S| - 1) & \text{if } \bar{y} \in \mathcal{C}_j^{\bar{y}} \\ |S| & \text{if } \bar{y} \in \mathcal{C}_j^e \setminus \mathcal{C}_j^{\bar{y}} \end{cases}$$

Since we are maximizing and  $\pi_{j,S} \leq 0$ , a variable may enter the basis if its reduced cost is nonpositive. Therefore, to find if an entering column belonging to  $\mathcal{C}_j^e$  exists, we should solve:

$$\max_{S \in \mathcal{C}_j^e} \sum_{i \in S} x_i - x_j + (|S| - 1 + \mathbb{1}_{S \in \mathcal{C}_j^e \setminus \mathcal{C}_j^{\bar{y}}}). \quad (8)$$

Here,  $\mathbb{1}_f$  is 1 if  $f$  is true and 0 otherwise. If the objective value is strictly positive, then the column  $\pi_{j,S} \in C_j^e$  should enter the LP. Problem (8) can be formulated as a MILP. We first rewrite the objective function as:

$$\sum_{i \in S} (x_i + 1) - x_j - 1 + \mathbb{1}_{S \in C_j^e \setminus C_j^{\bar{y}}}.$$

Our aim is to determine  $S \in C_j^e$ , which is a subset of the nodes in  $\delta^-(j)$ . We introduce binary variables  $\psi_i, i \in \delta^-(j)$  to select the nodes that will be included in  $S$ . We also need to determine if  $S \in C_j^e \setminus C_j^{\bar{y}}$ ; we use a binary variable  $\beta$  for this purpose. Furthermore, we only want to generate minimal activation sets: this is not necessary for correctness, but drastically reduces the set of columns. To model the minimality constraint, we use a decision variable  $\alpha = \min_{i \in \delta^-(j): \psi_i=1} w_{ij}$  that takes the value of the smallest arc weight within the chosen activation set. We then obtain the following pricing problem for node  $j \in V$ , where we use  $\bar{w}_j = \max_{i \in \delta^-(j)} w_{ij}$ :

$$\left. \begin{array}{l} -x_j - 1 + \max_{i \in \delta^-(j)} \sum_{i \in \delta^-(j)} (x_i + 1)\psi_i + \beta \\ \sum_{i \in \delta^-(j)} w_{ij}\psi_i \geq t_j \\ \sum_{i \in \delta^-(j)} (w_{ij} - \varphi_{ij}^{\bar{y}})\psi_i + (t_j + \theta_j^{\bar{y}})\beta \geq t_j + \theta_j^{\bar{y}} \\ \forall i \in \delta^-(j) \quad (\bar{w}_j - (w_{ij} - \varphi_{ij}^{\bar{y}}))\psi_i + \alpha \leq \bar{w}_j \\ \forall i \in \delta^-(j) \quad \sum_{i \in \delta^-(j)} (w_{ij} - \varphi_{ij}^{\bar{y}})\psi_i - \alpha \leq t_j + \theta_j^{\bar{y}} \\ \psi_i \in \{0, 1\} \\ \beta \in \{0, 1\} \\ \alpha \geq 0 \end{array} \right\} \text{(PRICE-}j\text{)}$$

When solving the pricing problem, one has to be careful in dealing with degeneracy. If the solution to the LP is degenerate, there can be columns with positive reduced cost that do not change the LP solution. This could lead to a situation in which we keep generating the same set of columns because the dual costs do not change, thereby entering an endless loop. To avoid this problem, we simply keep track of all the generated columns and make sure that we do not generate them again by adding no-good cuts in the pricing problem. For example, if  $\bar{S} \in C_j^e$  is an activation set generated by the pricing problem for node  $j \in V$ , then we include the no-good cut  $\sum_{i \notin \bar{S}} \psi_i + \sum_{i \in \bar{S}} (1 - \psi_i) \geq 1$  in all subsequent pricing problems solved for the same node  $j \in V$ . Finally, we remark that to check if all necessary columns have been generated, one has to verify that the value of (PRICE- $j$ ) is nonpositive for all  $j \in V$ .

The implementation of the algorithms described in this paper is based on the commercial software CPLEX, which does not allow adding columns at nodes of the Branch-and-Bound tree. Implementing an exact Branch-Cut-and-Price would require a different software framework. We decided instead to

implement a heuristic version of column generation, still relying on CPLEX. This is described in the next section.

## 6 Implementation of the algorithms

Sections 5.1 and 5.2 describe exact algorithms that would require fine control over the Branch-and-Cut process: an implementation of those algorithms must be able to change rows of the node LPs, and add columns after solving the pricing problems. These capabilities are not available when using commercial MILP solvers. Nonetheless, we implemented an exact algorithm, following Section 5.1, under the assumption that all columns (i.e., activation sets) can be generated in preprocessing, as well as a heuristic Branch-Cut-and-Price that periodically performs a pricing step, but does not offer optimality guarantees. The implementation is described here.

We start with the implementation of the exact algorithm under the assumption that all sets  $C_j^e$  are computed in a preprocessing step. We use this implementation on graphs of smaller size; it is called EXACT in Section 7. It works as follows.

- S1 Arbitrarily choose an initial  $\bar{y}$  to set the first row of (R-IMP- $\bar{y}$ ). Keep a lower bound  $L$  on the optimum; initially,  $L \leftarrow 0$ .
- S2 Solve (R-IMP- $\bar{y}$ ) in a Branch-and-Bound framework, using a callback to intercept any integer solution  $\tilde{y}$ . At every integer solution  $\tilde{y}$ , do:
  - Solve  $\text{RI}_{x,\theta,\varphi}(\tilde{y})$ .
  - If  $\text{RI}_{x,\theta,\varphi}(\tilde{y}) > L$ , save  $\tilde{y}$ , update  $L \leftarrow \text{RI}_{x,\theta,\varphi}(\tilde{y})$ , update  $\bar{y} \leftarrow \tilde{y}$  used to determine the first row of (R-IMP- $\bar{y}$ ), and restart, i.e., go back to [S2].
  - If  $\text{RI}_{x,\theta,\varphi}(\tilde{y}) < L$ , reject the incumbent and continue.
  - Else, if the value of the current incumbent matches the value of  $\text{RI}_{x,\theta,\varphi}(\tilde{y})$ , then accept the incumbent and continue; otherwise, reject the incumbent and continue.

Correctness of this algorithm is a direct consequence of the discussion in Section 5.1: at every restart, either the Branch-and-Bound finds a better solution, or it terminates proving optimality of the best known solution. In practice, we allow an exception to the rule that forces a restart: if  $\text{RI}_{x,\theta,\varphi}(\tilde{y}) > L$ , but we have processed more than  $10^6$  nodes in the Branch-and-Bound tree, or we have processed at least  $10^5$  nodes and the dual bound has improved compared to the root node, we do not restart. Instead, we continue the Branch-and-Bound until termination. The rationale is that we have made enough progress in the search, and restarting would be too detrimental.

Next, we describe the heuristic Branch-Cut-and-Price that periodically that we use on large graphs; this is called HEURCG in Section 7. The algorithm uses EXACT as a subroutine. It works as follows.

- Generate an initial set of columns  $C$ . Repeat the following until the time limit is reached:

- Launch EXACT using columns  $C$ , until it either finds a better solution  $\tilde{y}$  and is about to restart, or terminates proving optimality of a solution  $\tilde{y}$ .
- Perform a pricing step on the LP relaxation of (R-IMP- $\bar{y}$ ), fixing  $y = \tilde{y}$  and using  $\bar{y} = \tilde{y}$ . A pricing step consists of several pricing iterations, where each iteration solves (PRICE- $j$ ) for  $0.05|V|$  randomly chosen nodes  $j \in V$ . The pricing step performs a pricing iteration until: (i) the bound of the LP matches  $RI_{x,\theta,\varphi}(\tilde{y})$ , or (ii) we have performed  $2|V|$  pricing iterations, or (iii) we exceed a given maximum number of generated columns per pricing step.
- Let  $C'$  be the newly generated columns. Set  $C \leftarrow C \cup C'$ .

In our implementation, to generate the initial set of columns we apply REC-GENEXT to each node  $j \in V$  with an upper bound  $k$  on the cardinality of the activation sets: we start with  $k = 2$ , and iteratively increase  $k$  until we generate at least 5000 columns, keeping all columns generated this way (which may therefore be more than 5000).

## 7 Computational results

We test our approach on a large collection of graphs taken from the literature. All the proposed algorithms are written in Python using IBM ILOG CPLEX 12.8.0 as MILP solver. We activate the numerical emphasis setting because of some difficulties faced during our numerical evaluation (on some graphs, model (DUAL- $\theta$ - $\varphi$ 0) can have solutions with very large values). Related to this, we remark that reformulating the indicator constraints with big-M constraints is likely to fail: we tried a similar approach, but the big-M values required for validity are too large in practice. All the experiments reported in this section are executed on a homogeneous cluster equipped with Xeon E7-4850 processors (2.00GHz, 64GB RAM).

We consider graphs belonging to the class *SW* (“small world”), described in [22, 11]. We use two different test sets:

- *Small graphs.*  
Number of nodes  $n \in \{50, 75, 100\}$ , average node degree  $k \in \{4, 8\}$  for  $n = 50$  and  $k \in \{8, 12\}$  for  $n = 75, 100$ , and rewiring probability  $b \in \{0.1, 0.3\}$ . For each combination of settings, we used 5 random instances.
- *Large graphs.*  
Number of nodes  $n \in \{2000, 5000\}$ , average node degree  $k \in \{12, 16\}$  and rewiring probability  $b \in \{0.1, 0.3\}$ . For each combination of settings, we used 5 random instances.

Therefore, we have a total of 60 small instances and 40 large instances. We also tested a directed version of the Erdos-Rényi random graph [10]: the conclusions of the study are similar and are therefore not reported.

k	s	B <sub>N</sub>	time (s)			opt (frac)			gap (%)			Activation sets (#)		
			0	1	2	0	1	2	0	1	2	0	1	2
B <sub>A</sub>														
<hr/>														
4	3	0	3.4	11.4	22.9	1.0	1.0	1.0	0.0	0.0	0.0	119.2	156.4	177.6
		1	8.4	19.7	22.5	1.0	1.0	1.0	0.0	0.0	0.0	156.4	179.9	198.8
		2	11.8	19.0	22.0	1.0	1.0	1.0	0.0	0.0	0.0	176.4	198.1	211.2
<hr/>														
5	0	0	97.3	313.5	266.4	1.0	1.0	1.0	0.0	0.0	0.0	119.2	156.4	177.6
		1	292.4	287.3	360.1	1.0	1.0	1.0	0.0	0.0	0.0	156.4	179.9	198.8
		2	291.6	429.9	367.5	1.0	1.0	1.0	0.0	0.0	0.0	176.4	198.1	211.2
<hr/>														
8	0	0	1373.5	3231.4	3600.0	0.8	0.3	0.0	-27.1	35.0	55.0	119.2	156.4	177.6
		1	3204.8	3589.4	3600.0	0.3	0.1	0.0	34.8	47.9	67.4	156.4	179.9	198.8
		2	3545.2	3600.0	3600.0	0.1	0.0	0.0	48.0	60.2	67.3	180.0	198.1	211.2
<hr/>														
8	3	0	7.9	39.2	44.0	1.0	1.0	1.0	0.0	0.0	0.0	1025.4	1439.9	1806.1
		1	47.6	43.0	62.6	1.0	1.0	1.0	0.0	0.0	0.0	1439.9	1806.1	2123.3
		2	43.3	60.0	87.9	1.0	1.0	1.0	0.0	0.0	0.0	1806.1	2123.3	2392.7
<hr/>														
5	0	0	466.6	830.8	1404.4	1.0	1.0	0.9	0.0	0.0	6.4	1025.4	1439.9	1806.1
		1	767.9	1651.2	1922.6	1.0	0.8	0.8	0.0	24.4	20.7	1439.9	1806.1	2123.3
		2	1526.4	1501.7	1260.1	0.9	1.0	1.0	3.3	0.0	0.0	1806.1	2123.3	2392.7
<hr/>														
8	0	0	2880.6	3600.0	3600.0	0.2	0.0	0.0	47.0	79.7	72.1	1025.4	1439.9	1806.1
		1	3600.0	3600.0	3600.0	0.0	0.0	0.0	83.6	94.8	99.6	1439.9	1806.1	2123.3
		2	3600.0	3600.0	3600.0	0.0	0.0	0.0	82.3	84.2	139.0	1806.1	2123.3	2392.7

**Table 1** Exact algorithm on *Small graphs* with  $n = 50$ , averaged across instances of same average node degree  $k$ . Time limit 3600 s.

k	s	B <sub>N</sub>	time (s)			opt (frac)			gap (%)			Activation sets (#)		
			0	1	2	2	0	1	2	0	1	2	0	1
8	4	0	2427.9	3554.5	3595.9	0.8	0.1	0.0	26.4	153.3	194.6	1819.1	2642.4	3374.2
		1	3338.2	3600.0	3465.2	0.2	0.0	0.1	167.1	117.8	195.0	2642.4	3375.0	3994.6
		2	3596.0	3386.5	3505.9	0.0	0.2	0.1	98.9	96.7	163.3	3372.9	3993.4	4573.0
8	0	0	1805.5	3343.3	3600.0	0.5	0.1	0.0	61.9	90.0	100.0	1819.1	2642.4	3374.2
		1	3280.4	3600.0	3587.0	0.3	0.0	0.1	90.0	100.0	100.0	2642.4	3375.0	3994.6
		2	3600.0	3600.0	3600.0	0.0	0.0	0.0	100.0	100.0	100.0	3372.9	3993.4	4573.0
12	4	0	380.2	3288.2	3600.0	1.0	0.2	0.0	0.0	80.0	100.0	1819.1	2642.4	3374.2
		1	3254.9	3600.0	3600.0	0.1	0.0	0.0	90.0	100.0	100.0	2642.4	3375.0	3994.6
		2	3600.0	3600.0	3600.0	0.0	0.0	0.0	100.0	100.0	100.0	3372.9	3993.4	4573.0
12	4	0	3600.0	3600.0	3600.0	0.0	0.0	0.0	100.0	880.0	1085.0	21779.0	33850.8	44552.9
		1	3600.0	3599.4	3600.0	0.0	0.0	0.0	855.0	1215.0	787.5	33850.8	44553.3	54315.9
		2	3600.0	3600.0	3600.0	0.0	0.0	0.0	1047.5	1085.0	1532.5	44552.9	54315.9	63020.0
8	0	0	3248.7	3578.9	3600.0	0.4	0.1	0.0	100.0	155.0	155.0	21779.0	33850.8	44552.9
		1	3600.0	3600.0	3600.0	0.0	0.0	0.0	210.0	281.7	248.3	33850.8	44553.3	54315.9
		2	3600.0	3584.7	3600.0	0.0	0.1	0.0	218.3	303.3	416.7	44552.9	54315.9	63020.0
12	0	0	1810.0	2882.3	3301.6	0.5	0.2	0.1	100.0	80.0	90.0	21779.0	33850.8	44552.9
		1	2201.5	3272.2	3333.3	0.4	0.1	0.1	60.0	90.0	140.0	33850.8	44553.3	54315.9
		2	3600.0	3287.1	3600.0	0.0	0.1	0.0	121.7	144.1	100.0	44552.9	54315.9	63020.0

**Table 2** Exact algorithm on *Small graphs* with  $n = 75$ , averaged across instances of same average node degree  $k$ . Time limit 3600 s.



k	s	B <sub>N</sub>	time (s)			opt (frac)			gap (%)			Activation sets (#)		
			0	1	2	0	1	2	0	1	2	0	1	2
8	5	0	2754.5	3600.0	3582.8	0.4	0.0	0.1	282.2	174.4	181.4	2374.3	3363.3	4247.4
		1	3600.0	3598.3	3600.0	0.0	0.0	0.0	169.0	261.4	90.2	3363.3	4249.3	5027.6
		2	3600.0	3600.0	3600.0	0.0	0.0	0.0	87.6	182.5	171.8	4247.0	5027.3	5694.7
10	0	0	1402.0	3600.0	3600.0	0.8	0.0	0.0	5.7	66.2	76.6	2374.3	3363.3	4247.4
		1	3369.0	3600.0	3600.0	0.2	0.0	0.0	65.8	77.7	80.5	3363.3	4249.3	5027.6
		2	3600.0	3600.0	3600.0	0.0	0.0	0.0	71.8	79.0	80.8	4247.0	5027.3	5694.7
15	0	0	352.3	3355.3	3600.0	1.0	0.1	0.0	40.0	48.7	51.5	2374.3	3363.3	4247.4
		1	3587.7	3600.0	3584.0	0.1	0.0	0.1	54.6	62.3	72.1	3363.3	4249.3	5027.6
		2	3600.0	3600.0	3600.0	0.0	0.0	0.0	65.7	71.8	73.1	4247.0	5027.3	5694.7
12	5	0	3600.0	3600.0	3600.0	0.0	0.0	0.0	1423.8	1158.0	830.2	28988.7	44675.0	58758.9
		1	3590.8	3600.0	3583.7	0.1	0.0	0.1	682.8	977.6	1010.8	44675.0	58758.9	71234.1
		2	3600.0	3600.0	3578.7	0.0	0.0	0.2	1305.5	1010.9	1291.5	58758.9	71234.1	82334.8
10	0	0	1842.4	2578.1	3242.8	0.5	0.3	0.1	274.7	103.6	190.8	28988.7	44675.0	58758.9
		1	2564.1	3242.3	3281.4	0.4	0.1	0.1	103.6	126.1	65.9	44675.0	58758.9	71234.1
		2	3249.0	3237.4	3268.4	0.1	0.2	0.1	124.7	124.9	151.5	58758.9	71234.1	82334.8
15	0	0	1772.1	1815.4	2002.8	0.6	0.5	0.5	71.4	38.5	39.1	28988.7	44675.0	58758.9
		1	1818.3	1989.8	2226.6	0.5	0.5	0.5	63.1	39.0	39.7	44675.0	58758.9	71234.1
		2	1877.8	1979.9	2972.7	0.6	0.5	0.3	39.2	39.7	40.0	58758.9	71234.1	82334.8

**Table 3** Exact algorithm on *Small graphs* with  $n = 100$ , averaged across instances of same average node degree  $k$ . Time limit 3600 s.

We solve the robust and non-robust IMP on each graph using different sets of parameters and algorithms. The number of seeds is chosen as a fraction (rounded up to an integer)  $s \in \{0.05, 0.10, 0.15\}$  of  $n$  for the *Small Graphs*. The robustness budgets  $B_N$  and  $B_A$  are chosen from  $\{0, 1, 2\}$ , where the case with  $B_N = B_A = 0$  corresponds to the non-robust case (i.e., the deterministic linear threshold model). We use  $s \in \{0.10, 0.15\}$ ,  $B_N, B_A \in \{0, 25, 50\}$  for the *Large Graphs*. The maximum deviations  $\Delta_N, \Delta_A$  are always equal to 0.1.

Table 1, 2 and 3 illustrate the results obtained on *Small Graphs* with EXACT, where all extended collections of activation sets are generated in a pre-processing phase with Alg. 2. The tables report the average computation time, the average gap, the fraction of instances solved to optimality and the average number of activation sets generated for each combination of parameters. In our tests CPLEX generates very few cuts – mostly implied bounds, but almost no other cut. This may be related to the numerical difficulties mentioned above. The robust version seems harder than the non-robust counterpart, but at least for smaller instances the computation time is similar: for  $n = 50$ , whenever we can solve all deterministic instances, we can solve the corresponding robust problems as well. The complexity of the instance increases rapidly with the number of nodes and with the number of seeds; this is not surprising and agrees with similar findings presented in [11]. The number of activation sets is strongly dependent on  $k$ ; already for  $n = 100, k = 12$  there are several thousand activation sets, indicating that column generation is necessary to scale up the size. The fact that we are able to solve to optimality around 15% of the instances with  $n = 100$  shows that this problem is very difficult even at modest sizes.

On *Large Graphs*, we compare HEURCG with a multistart 2-OPT heuristic. 2-OPT works as follows: it randomly chooses a set of seed nodes  $S \subset V$  of the desired cardinality; then, it explores a neighborhood centered at  $S$ , defined as  $(S \setminus \{u\}) \cup \{v\}$ , where  $u \in S$  and  $v \in V \setminus S$ , trying all possible such  $u$  and  $v$ . If an improving solution is found in the neighborhood, it accepts the new solution, and repeats the neighborhood search from the new  $S$ . If there is no better solution, it restarts by generating a new random  $S$ . This is repeated until the time limit is hit. The results reported here are for a 2-OPT in which the neighborhood is always fully explored before moving to a new solution, so as to choose the largest improvement; we also tested an alternative version in which an improving move is accepted immediately after it is discovered, but its performance was slightly worse.

Table 4 reports results obtained with the two heuristics. It shows the average value of the best incumbent found within one hour of time limit for each combination of parameters. HEURCG clearly outperforms 2-OPT in this metric. Additionally, we report that HEURCG finds a better solution on 720 out of 720 instances, and is thus significantly more effective. In fact, 2-OPT typically finds a solution that attains influence only marginally larger than the number of seed nodes, while HEURCG achieves nontrivial improvements over that value.

n	k	Seeds	BA	0		25		50	
			BN	HEURCG	2opt	HEURCG	2opt	HEURCG	2opt
2000	12	200	0	234.0	201.7	211.1	200.4	210.3	200.5
			25	210.6	200.7	202.8	200.1	202.6	200.1
			50	209.9	200.6	202.8	200.2	202.8	200.1
	16	200	0	239.4	200.5	207.7	200.1	208.1	200.0
			25	212.9	200.1	202.9	200.1	203.8	200.0
			50	212.5	200.1	203.7	200.1	203.8	200.0
	16	300	0	373.4	301.6	323.3	300.9	323.6	301.1
			25	327.0	300.9	309.7	300.8	306.7	300.5
			50	327.2	301.1	306.8	300.3	305.8	300.5
5000	12	500	0	586.8	501.7	532.7	500.6	527.3	500.6
			25	535.4	500.5	521.2	500.6	515.9	500.4
			50	528.9	500.6	517.6	500.6	511.6	500.5
	16	500	0	554.8	500.2	528.6	500.0	524.9	500.0
			25	530.2	500.1	521.3	500.0	517.2	500.0
			50	526.0	500.1	519.4	500.0	516.7	500.0
	16	750	0	832.1	750.8	811.3	750.0	801.4	750.0
			25	816.4	750.3	797.3	750.0	789.5	750.0
			50	804.0	750.3	789.6	750.0	783.4	750.0

**Table 4** Heuristic Comparison: best incumbent for  $SW$  graphs with  $n = 2000, 5000$ , averaged across instances of same average node degree  $k$ .

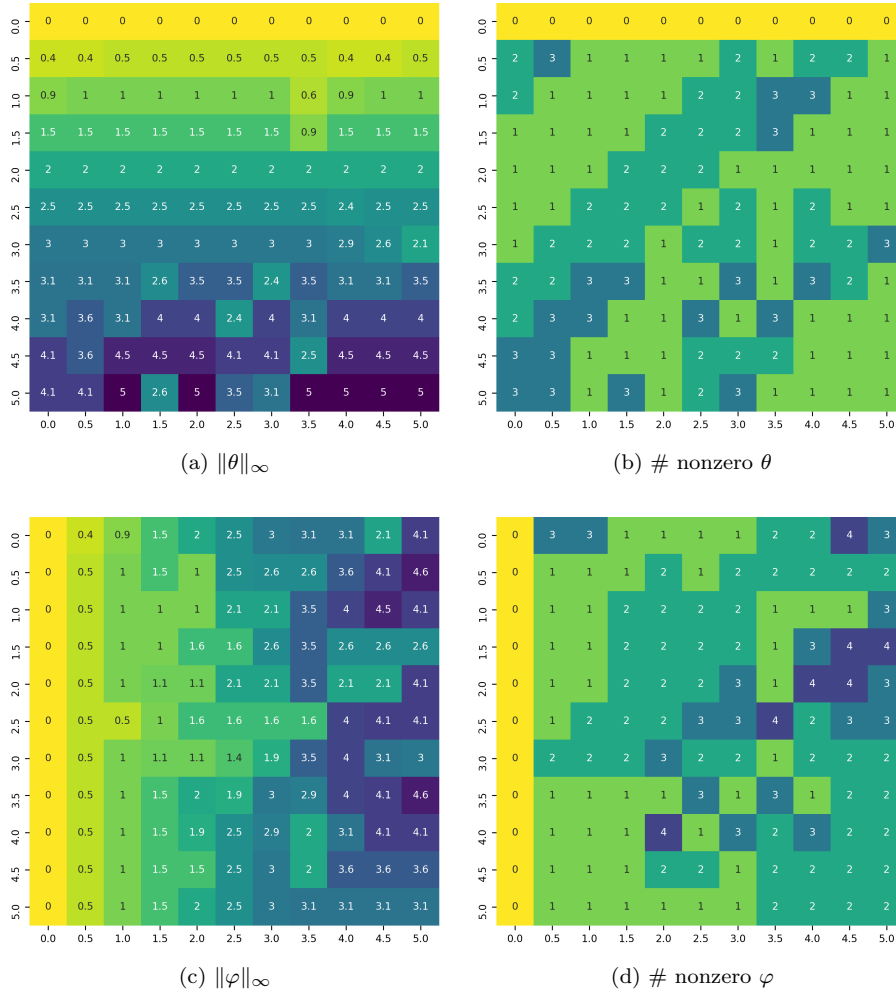
Looking at the relation between the values of the best solution and the robustness parameters, we see that the price of robustness is quite high and protecting against a small amount of uncertainty decreases significantly the total number of activated nodes. We analyze this in more detail in the next subsection.

### 7.1 The price of robustness: a case study

Our results in Section 3.2 indicate that the price to pay to protect the solution from changes in the node thresholds should be at least as high as the one for changes in the arc weights. We performed a detailed study to verify the empirical behavior on our test set. We looked at a few of these cases; here we

report only one, as the conclusions are similar — although we did not verify if the entire set of test instances exhibits the same pattern.

We study a *SW* graph with the following characteristics:  $n = 50$ ,  $k = 4$ ,  $b = 0.1$ . We examine how the robustness budget affects the value of the optimum. We choose a range of values for  $B_N$  and  $B_A$  from 0 to 5, with an increment of 0.5, for a total of 121 possible combinations. We set  $\Delta_A = 1$  and  $\Delta_N = \frac{B_N}{\min_j t_j}$ , so that no upper bounds on the values of  $\theta$  and  $\varphi$  are imposed.



**Fig. 3** The maximum values of  $\theta$  and  $\varphi$  are reported in (a) and (c). The number of nonzeros of  $\theta$  and  $\varphi$  are reported in (b) and (d). In each subfigure, the value of  $B_N$  increases from top to bottom and the value of  $B_A$  increases from left to right.

In Fig. 3 we report the maximum value of the  $\theta$  and  $\varphi$  variables, i.e., the  $\ell_\infty$ -norm of the corresponding vector, as well as the number of nonzero  $\theta$  and  $\varphi$  variables at the optimum. Each subtable shows the values of  $\|\theta\|_\infty$ ,  $\|\varphi\|_\infty$ ,  $\#$  nonzero  $\theta$  and  $\#$  nonzero  $\varphi$  for increasing values of  $B_A$  from right to left and for increasing values of  $B_N$  from top to bottom.

The values of  $\|\theta\|_\infty$  increase as  $B_N$  increases. In the majority of the cases  $\|\theta\|_\infty$  is equal to  $B_N$ , and a similar observation holds for  $\|\varphi\|_\infty$  and  $B_A$ . As a consequence, there are typically few nonzero  $\theta$  and  $\varphi$ , indicating that the worst-case realization of uncertainty is represented by a situation in which very few nodes/arcs are severely affected. In Fig. 4 we show the change in the optimal value as  $B_N$  and  $B_A$  increase. The figure shows that a small amount of robustness is sufficient to significantly decrease the amount of influence attained. The change in the optimal values is remarkably symmetric in  $B_N$  and  $B_A$ .

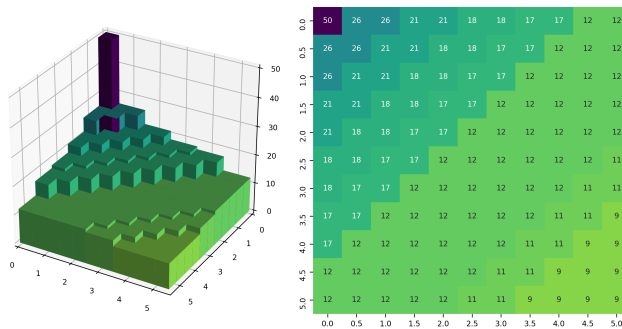


Fig. 4 Optimal value of the problem for different values of  $B_N$  and  $B_A$ .

## 8 Conclusions

We present an exact Branch-and-Cut and a heuristic Branch-Cut-and-Price for the influence maximization problem, and its robust counterpart. The algorithms for the robust version of the problem can handle uncertainty on the edge weights and on the nodes thresholds. To solve the deterministic version of the problem we introduce a new formulation originating from a bilevel problem, exploiting the dual of some (integral) subproblems. The formulation is extended to the robust case with a row generation and replacement scheme, within a Branch-and-Cut framework. Since the number of decision variables in our formulation is exponential, we discuss a column generation scheme, and implement a heuristic Branch-Cut-and-Price. Numerical experiments show that we are able to solve to optimality the robust and the deterministic version of the problem on small graphs; instances with 100 or more nodes are challenging

already. However, we test our heuristic on graphs with up to 5000 nodes and find that it significantly outperforms a simple 2-opt heuristic.

## Acknowledgments

Part of this research was conducted with support by the Simons Foundation and by the Mathematisches Forschungsinstitut Oberwolfach.

## References

1. Ackerman, E., Ben-Zwi, O., Wolfowitz, G.: Combinatorial model and bounds for target set selection. *Theoretical Computer Science* **411**(44-46), 4017–4022 (2010)
2. Banerjee, S., Jenamani, M., Pratihari, D.K.: A survey on influence maximization in a social network. arXiv preprint arXiv:1808.05502 (2018)
3. Ben-Tal, A., Nemirovski, A.: Selected topics in robust convex optimization. *Mathematical Programming* **112**(1), 125–158 (2008)
4. Bertsimas, D., Brown, D.B., Caramanis, C.: Theory and applications of robust optimization. *SIAM review* **53**(3), 464–501 (2011)
5. Bertsimas, D., Sim, M.: The price of robustness. *Operations research* **52**(1), 35–53 (2004)
6. Buchheim, C., Kurtz, J.: Robust combinatorial optimization under convex and discrete cost uncertainty. *EURO Journal on Computational Optimization* **6**(3), 211–238 (Sep 2018)
7. Chen, W., Lin, T., Tan, Z., Zhao, M., Zhou, X.: Robust influence maximization. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 795–804. ACM (2016)
8. Domingos, P., Richardson, M.: Mining the network value of customers. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 57–66. KDD '01, ACM, New York, NY, USA (2001)
9. Edmonds, J., Giles, R.: A min-max relation for submodular functions on graphs. In: *Annals of Discrete Mathematics*, vol. 1, pp. 185–204. Elsevier (1977)
10. Erdos, P., Rényi, A.: On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci* **5**(1), 17–60 (1960)
11. Fischetti, M., Kahr, M., Leitner, M., Monaci, M., Ruthmair, M.: Least cost influence propagation in (social) networks. *Mathematical Programming* **170**(1), 293–325 (Jul 2018)
12. Goldberg, S., Liu, Z.: The diffusion of networking technologies. In: Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013. pp. 1577–1594 (2013)
13. Gunnec, D.: Integrating social network effects in product design and diffusion. Ph.D. thesis (2012)
14. He, X., Kempe, D.: Stability and robustness in influence maximization. *ACM Transactions on Knowledge Discovery from Data (TKDD)* **12**(6), 66 (2018)
15. Kempe, D., Kleinberg, J., Tardos, E.: Maximizing the spread of influence through a social network. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 137–146. KDD '03, ACM, New York, NY, USA (2003)
16. Kempe, D., Kleinberg, J., Tardos, E.: Maximizing the spread of influence through a social network. *Theory of Computing* **11**(4), 105–147 (2015)
17. Könemann, J., Sadehghabadi, S.S., Sanità, L.: Better approximation algorithms for technology diffusion. In: Algorithms - ESA 2013 - 21st Annual European Symposium, Sophia Antipolis, France, September 2-4, 2013. Proceedings. pp. 637–646 (2013)
18. Li, Y., Fan, J., Wang, Y., Tan, K.L.: Influence maximization on social graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering* (2018)

19. Li, Z., Ding, R., Floudas, C.A.: A comparative theoretical and computational study on robust counterpart optimization: I. Robust linear optimization and robust mixed integer linear optimization. *Industrial & engineering chemistry research* **50**(18), 10567–10603 (2011)
20. Mossel, E., Roch, S.: On the submodularity of influence in social networks. In: *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*. pp. 128–134. ACM (2007)
21. Richardson, M., Domingos, P.: Mining knowledge-sharing sites for viral marketing. In: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. pp. 61–70. ACM (2002)
22. Watts, D.J., Strogatz, S.H.: Collective dynamics of "small-world" networks. *Nature* **393**(6684), 440 (1998)
23. Wu, H.H., Küçükyavuz, S.: A two-stage stochastic programming approach for influence maximization in social networks. *Computational Optimization and Applications* **69**(3), 563–595 (2018)