Mathematical Programming (2020) 183:419–453
https://doi.org/10.1007/s10107-020-01507-z

FULL LENGTH PAPER

Series B

An exact algorithm for robust influence maximization

Giacomo Nannicini[1] · Giorgio Sartor[2] · Emiliano Traversi[3] · Roberto Wolfler Calvo[3,4]

Received: 14 June 2019 / Accepted: 10 April 2020 / Published online: 19 May 2020

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 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.

✉ Giacomo Nannicini
nannicini@us.ibm.com

Giorgio Sartor
giorgio.sartor@sintef.no

Emiliano Traversi
traversi@lipn.fr

Roberto Wolfler Calvo
wolfler@lipn.fr

1 IBM T.J. Watson Research Center, Yorktown Heights, NY, USA

2 SINTEF Digital, Oslo, Norway

3 Université Sorbonne Paris Nord, LIPN,CNRS, UMR 7030, 93430 Villetaneuse, France

4 Department of Mathematics and Computer Science, University of Cagliari, Cagliari, Italy

Keywords Influence maximization · Mixed-integer programming · Robust optimization

Mathematics Subject Classification 90B10 · 90C11 · 90C17

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 Domingos and Richardson [9] and Richardson and Domingos [23]. Several variants of the IMP have been presented in the literature; we refer to the recent surveys Banerjee et al. [2], Kempe et al. [17] and Li et al. [19] 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., Kempe et al. [16]. 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., Ben-Tal and Nemirovski [4], Bertsimas et al. [6], Buchheim and Kurtz [7], Li et al. [20]). 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 Mossel and Roch [22]. Hence, it admits an efficient $(1 - \frac{1}{e})$-approximation using a greedy algorithm, see e.g., Kempe et al. [16], Mossel and Roch [22]. 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 Chen et al. [8] for an approximation algorithm for a robust version of IMP under some conditions, that still requires submodularity).

Kempe et al. [16,17] study the greedy approach for the threshold and cascade model in the stochastic setting. Wu and Küçükyavuz [26] proposes an exact cutting plane approach for the same class of models, using strong optimality cuts exploiting submodularity. Chen et al. [8], He and Kempe [15] 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) Ackerman et al. [1], the (Generalized) Least Cost Influence Problem (GLCIP) Fischetti et al. [12], Gunnec [14], and the Technology Diffusion Problem (TDP) Goldberg and Liu [13], Könemann et al. [18]. 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 Ackerman et al. [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 Fischetti et al. [12]. Both Ackerman et al. [1] and Fischetti et al. [12] 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. Goldberg and Liu [13] propose 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 Könemann et al. [18] 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 Γ-robustness of Bertsimas and Sim [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 Kempe et al. [17]; 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 Bertsimas and Sim [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 Bertsimas and Sim [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 Fischetti et al. [12]. We show that our integer programming formulation for the non-robust model is competitive with the exact algorithm of Fischetti et al. [12] 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 Sect. 8 concludes the paper.

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

```
1: $A_0 \leftarrow \{j \in V : y_j = 1\}$
2: for $k = 1, \ldots, n$ 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).

$\delta^-(j)$ 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.

Definition 1 Given seeds y, a vector of nonnegative node thresholds t, and nonnegative arc weights w, we define the set of *active nodes* as the set returned by Algorithm 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. A summary of this notation, as well as all the notation used in this paper, can be found in "Appendix A".

The activation function used in Algorithm 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 Δ_N from its nominal value, and the total amount of threshold variations is upper bounded by a given number B_N. Similary, each arc weight w_{ij} can deviate by some fraction Δ_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:

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

where a tight big-M coefficient can be computed as $\sum_{i \in \delta^-(j)} w_{ij} + \epsilon$. In the above formulation, ϵ 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. (In our numerical experiments, the integrality tolerance used by the solver is several orders of magnitude smaller than this value.)

For each node j, we use variable θ_j to denote the increase of node threshold t_j, and for each arc (i, j), we use variable φ_{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 Bertsimas and Sim [5]. We need the binary variable x_j in the upper bound for φ_{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 θ_j and arc weight change φ_{ij}, problem $\text{RI}_{x,\theta,\varphi}(\bar{y})$ 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 θ and φ, thereby yielding the influence in the worst-case realization of uncertainty.

Notice that $\text{RI}_{x,\theta,\varphi}(\bar{y})$ 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 $\sum_{i \in \delta^-(j)}(w_{ij}x_i - \varphi_{ij}) \geq t_j + \theta_j$. Notice that, because of the constraints $\varphi_{ij} \leq \Delta_N x_i$, this expression is equivalent to $\sum_{i \in \delta^-(j)}(w_{ij} - \varphi_{ij})x_i \geq t_j + \theta_j$, 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 $j \in A_n$. □

The above formulation does not suffer from self-activating loops (which require additional caution in some other formulations, see e.g., Fischetti et al. [12]), and it has n binary variables only. Problem $\text{RI}_{x,\theta,\varphi}(\bar{y})$ is stated as a MILP; Sect. 3.1 shows that computing its value is NP-hard, hence using a MILP formulation is justified. Because of Proposition 1, the robust IMP with q activation seeds can be solved as the following bilevel optimization problem:

$$\left.\begin{array}{lrl} & \max_y \; \text{RI}_{x,\theta,\varphi}(y) & \\ & \sum_{j \in V} y_j & = q \\ \forall j \in V & y_j & \in \{0, 1\}. \end{array}\right\} \qquad \text{(R-IMP)}$$

Notice that if we fix $\theta = 0$ and $\varphi = 0$, solving $\text{RI}_{x,\theta=0,\varphi=0}(\bar{y})$ 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 θ and φ. The robust approach of Bertsimas and Sim [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 Algorithm 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 Sects. 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 $\mathrm{RI}_{x,\theta,\varphi}(\bar{y})$ 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 $\mathrm{RI}_{x,\theta,\varphi}(y)$ 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 Δ_N, Δ_A, and an integer k.
Output: is $\mathrm{RI}_{x,\theta,\varphi}(\bar{y}) \leq k$?

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 $\textsc{InfluenceSpread}(\bar{y}, t + \theta, w + \varphi)$. 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 th 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 $j \in C$; 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 $(1 + \epsilon)$. □

3.2 Sensitivity analysis

Given a set of robustness parameters $P = \{B_N, \Delta_N, B_A, \Delta_A\}$, we denote by $\mathrm{RI}^P_{x,\theta,\varphi}(y)$ the problem $\mathrm{RI}_{x,\theta,\varphi}(y)$ 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 Δ_N, Δ_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* $P^1 = \{B^1_N, \Delta^1_N = +\infty, B^1_A, \Delta^1_A = 1\}$, $P^2 = \{B^2_N, \Delta^2_N = +\infty, B^2_A, \Delta^2_A = 1\}$. *We have the following implication:*

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

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

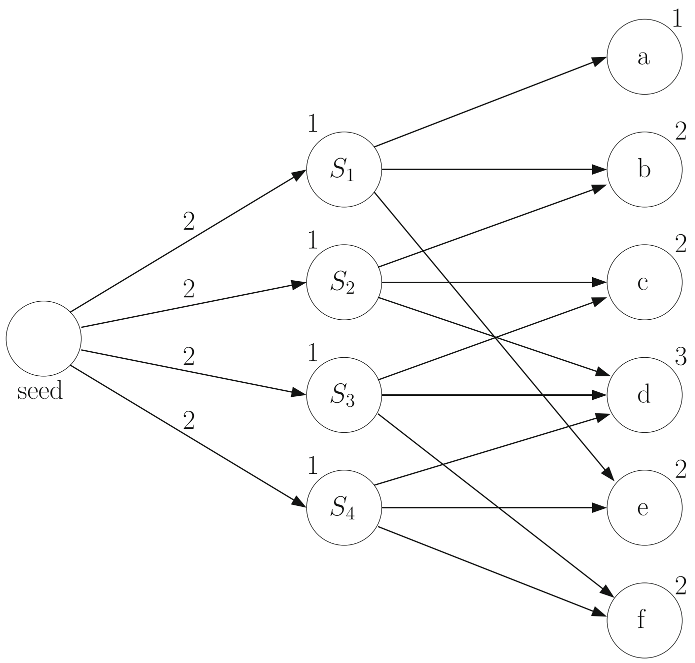


Fig. 1 Example of the reduction from "Exact Cover by 3-Sets." The ground set is $X = \{a, b, c, d, e, f\}$, $q = 2$, $\mathcal{C} = \{S_1, S_2, S_3, S_4\}$ with $S_1 = \{a, b, e\}$, $S_2 = \{b, c, d\}$, $S_3 = \{c, d, f\}$, $S_4 = \{d, e, f\}$. Node thresholds are indicated next to each node, and arc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

feasible for $\mathrm{RI}^{P_2}_{x,\theta,\varphi}(y)$ as follows:

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

It is immediate to check that $\mathrm{RI}^{P^1}_{x,\theta,\varphi}(y) = \mathrm{RI}^{P^2}_{x,\theta,\varphi}(y)$, and that the point satisfies the constraints $0 \le \theta_j^2$, $0 \le \varphi_{ij}^2$ and $\sum_{i\in\delta^-(j)}(w_{ij}x_i^2 - \varphi_{ij}^2) - \theta_j^2 + \epsilon - (\sum_{i\in\delta^-(j)} w_{ij} + \epsilon)x_j^2 \le t_j$. □

To prove Proposition 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 $\theta \le \Delta_N t$ and $\varphi \le \Delta_A w x$ in the definition of $\mathrm{RI}_{x,\theta,\varphi}(y)$. The next two results give sufficient conditions on Δ_N and Δ_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* $P^1 = \{B^1_N, \Delta^1_N, B^1_A, \Delta^1_A\}$*,* $P^2 = \{B^2_N, \Delta^2_N, B^2_A, \Delta^2_A\}$*. We have the following implication:*

$$\begin{cases} \Delta^2_N \geq \Delta^1_N + \frac{B^1_A - B^2_A}{B^1_A}\Delta^1_A \max_{j\in V}\{\frac{\sum_{i\in\delta^-(j)} w_{ij}}{t_j}\} \\ B^1_N + B^1_A = B^2_N + B^2_A \\ B^1_N \leq B^2_N \end{cases} \implies RI^{P^1}_{x,\theta,\varphi}(y) \geq RI^{P^2}_{x,\theta,\varphi}(y).$$

Proof As for Proposition 2, we show that for any feasible solution of $\mathrm{RI}^{P_1}_{x,\theta,\varphi}(y)$ we can construct a feasible solution for $\mathrm{RI}^{P_2}_{x,\theta,\varphi}(y)$ 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 Eq. (2). We have to show that the upper bounding constraints in $\mathrm{RI}^{P_2}_{x,\theta,\varphi}(y)$ hold for the proposed solution $(x^2, \theta^2, \varphi^2)$. We have:

$$\sum_{j\in V}\theta^2_j = \sum_{j\in V}\theta^1_j + \frac{B^1_A - B^2_A}{B^1_A}\sum_{j\in V}\sum_{i\in\delta^-(j)}\varphi^1_{ij} \leq B^1_N + \frac{B^1_A - B^2_A}{B^1_A}B^1_A = B^2_N$$

and

$$\varphi^2_{ij} = \frac{B^2_A}{B^1_A}\varphi^1 \leq \frac{B^2_A}{B^1_A}B^1_A = B^2_A.$$

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

$$\frac{\theta^2_j}{t_j} = \frac{\theta^1_j}{t_j} + \frac{B^1_A - B^2_A}{B^1_A}\sum_{i\in\delta^-(j)}\frac{\varphi^1_{ij}}{t_j} \leq \Delta^1_N + \frac{B^1_A - B^2_A}{B^1_A}\sum_{i\in\delta^-(j)}\frac{\Delta^1_A w_{ij} x_{ij}}{t_j} \leq \Delta^2_N.$$

Thus, θ^2 satisfies the constraints $\theta^2_j \leq \Delta^2_N t_j$. □

Theorem 3 *Let* $P^1 = \{B^1_N, \Delta^1_N, B^1_A, \Delta^1_A\}$*,* $P^2 = \{B^2_N, \Delta^2_N, B^2_A, \Delta^2_A\}$*. We have the following implication:*

$$\begin{cases} \Delta^2_N \geq \Delta^1_N + \frac{B^1_A - B^2_A}{B^1_A}\frac{1}{1-\Delta^1_A} \\ B^1_N + B^1_A = B^2_N + B^2_A \\ B^1_N \leq B^2_N \end{cases} \implies RI^{P^1}_{x,\theta,\varphi}(y) \geq RI^{P^2}_{x,\theta,\varphi}(y).$$

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

$$x^2 = x^1;\ \ \varphi^2 = \frac{B^2_A}{B^1_A}\varphi^1;\ \ \theta^2_j = \begin{cases} \theta^1_j + \frac{B^1_A - B^2_A}{B^1_A}\sum_{i\in\delta^-(j)}\varphi^1_{ij} & \text{if } x^1_j = 1 \\ 0 & \text{if } x^1_j = 0 \end{cases},\ \forall j \in V\ .$$

It is immediate to check that $\mathrm{RI}^{P1}_{x,\theta,\varphi}(y) = \mathrm{RI}^{P2}_{x,\theta,\varphi}(y)$, and that all the constraints of $\mathrm{RI}^{P2}_{x,\theta,\varphi}(y)$ except for $\theta^2_j \leq \Delta^2_N t_j$ are respected. Since $(x^1, \theta^1, \varphi^1)$ is feasible, we have $\varphi^1 \leq \Delta^1_A w x^1_i \leq \Delta^1_A w$, which implies:

$$(1 - \Delta^1_A) \sum_{i \in \delta^-(j)} w_{ij} \leq \sum_{i \in \delta^-(j)} (w_{ij} - \varphi^1_{ij}).$$

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

$$\frac{t_j}{(1 - \Delta^1_A)} \geq \sum_{i \in \delta^-(j)} w_{ij} \geq \sum_{i \in \delta^-(j)} \varphi^1_{ij},$$

where we used the fact that $\varphi^1_{ij} \leq \Delta^1_A w_{ij} x^1_i \leq \Delta^1_A w_{ij}$ and $0 \leq \Delta_A \leq 1$. Finally, this implies:

$$\frac{\theta^2_j}{t_j} = \frac{\theta^1_j}{t_j} + \frac{B^1_A - B^2_A}{B^1_A} \sum_{i \in \delta^-(j)} \frac{\varphi^1_{ij}}{t_j} \leq \Delta^1_N + \frac{B^1_A - B^2_A}{B^1_A} \sum_{i \in \delta^-(j)} \frac{\varphi^1_{ij}}{t_j} \leq \Delta^2_N.$$

□

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:*

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

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

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

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. The formulation that we use relies on the following definition:

Definition 2 For all $j \in V$, we define the corresponding *collection of minimal activation sets* $\mathcal{C}_j$ as:

$$\mathcal{C}_j = \left\{ S \subseteq \delta^-(j) : \sum_{i \in S} w_{ij} \geq t_j, S \text{ is minimal} \right\}$$

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. The concept of minimal activation set was first introduced in the recent paper Fischetti et al. [12]. 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 Fischetti et al. [12] in the context of the linear threshold model and no incentives. We can reformulate $\text{RI}_{x,\theta=0,\varphi=0}(\bar{y})$ (see Eq. (1)) using the collection of minimal activation sets.

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

Proposition 3 *If $\bar{y}$ is a 0–1 vector, the optimal solution x^* to $AS_x(\bar{y})$ is integer and $x_j = 1$ if and only if $j \in A_n$ as returned by* 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 $k = 1, \ldots, n$ that for every node $j \in A_k$ in 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 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 $j \in A_n$. □

We can use $\mathrm{AS}_x(\bar{y})$ 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:

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

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

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

The solution of this problem has value equal to that of its primal problem and therefore, by Proposition 3, to InfluenceSpread($\bar{y}, t, w$) whenever $\bar{y} \in \{0,1\}^n$. It follows that a valid formulation for (IMP-θ0-φ0) is the following:

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

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 $y_j = 0$, μ_j has no contribution in the objective function; since μ_j appears in a single constraint and increasing μ_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-θ0-φ0):

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

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.

An alternative approach to deal with bilinear terms is to linearize them via McCormick envelopes (see McCormick [21]), which are exact when y is binary and μ is bounded. It is also possible to explicitly model the logical implication

$y_j = 0 \Rightarrow \mu_j = 0$ with a big-M constraints. With these last two approaches it is necessary to find valid upper bounds for the μ variables, since these upper bounds appear explicitly in the linearizations. Unfortunately, the required upper bounds are large: in "Appendix B" we show that n is not a valid bound, and numerically we found cases where even n^2 was not enough. Hence, explicit computation of upper bounds of μ seems difficult and would in any case lead to weak formulations. We note, however, that in a Branch-and-Bound framework the upper bounds could be iteratively tightened, see, e.g., Belotti et al. [3].

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-$\theta 0$-$\varphi 0$) is an integral polyhedron.*

Proof We show that for a given 0-1 vector $\bar{y}$, the remaining system in (DUAL-$\theta 0$-$\varphi 0$) is total dual integral. This implies that it defines an integral polyhedron Edmonds and Giles [10].

The discussion in this section shows that the dual of (DUAL-$\theta 0$-$\varphi 0$) for fixed $y = \bar{y}$ is the problem $\text{AS}_x(\bar{y})$ 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 $\text{AS}_x(\bar{y})$, either $\text{AS}_x(\bar{y})$ 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 $k = 1, \ldots, n$, (i) set $x^k_j \leftarrow 0 \, \forall j$; (ii) for $j \in V, S \in \mathcal{C}_j$, set $x^k_j \leftarrow \max\{x^k_j, \sum_{i \in S} x^{k-1}_i - b_{j,S}\}$. 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 $x^k_h > x^*_h$ for some h. Since x^k_h is initially 0, it must be that for some S, x^k_h is set to $\sum_{i \in S} x^{k-1}_i - b_{h,S} > x^*_h$ for the first time. But $\sum_{i \in S} x^{k-1}_i - b_{h,S} \leq \sum_{i \in S} x^*_i - b_{h,S} \leq x^*_h$, because x^* satisfies the constraints; this is a contradiction. It follows that $x^k \leq x^*$ for all $k = 1, \ldots, n$. It is easy to check that x^n is feasible by construction, and therefore it must be optimal. □

The proof of Proposition 4 shows that the LP obtained from (DUAL-$\theta 0$-$\varphi 0$) for integral y is total dual integral. Hence, variables π, μ are automatically integer when y is integer. However, empirically we did not observe any advantage by imposing π, μ 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 θ and φ as in the original formulation (R-IMP), but it is not obvious how to use the dualization trick described in the previous section when θ and φ are not fixed. This is because the activation sets may change with θ and φ, since θ directly affects the node activation thresholds, while φ affects the edges values. To overcome this difficulty, we propose to work with a modification of (DUAL-θ0-φ0) that includes dual $\pi_{j,S}$ variables for all the activation sets that may be minimal for any of the possible values of θ and φ. 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 θ and φ. 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 $\mathrm{RI}_{x,\theta,\varphi}(\bar{y})$, i.e., the maximum influence spread when simultaneously minimizing over θ and φ.

It will be convenient to introduce the following sets.

Definition 3 For all $j \in V$ and robustness parameters Δ_N and Δ_A, we define the corresponding *extended collection of minimal activation sets* $\mathcal{C}^e_j$ as:

$$\mathcal{C}^e_j = \Bigg\{ S \subseteq \delta^-(j) : \exists\, \theta_j \in [0, \Delta_N t_j], \varphi_{ij} \in [0, \Delta_A w_{ij}] \text{ such that } \sum_{i \in S} (w_{ij} - \varphi_{ij}) \geq t_j + \theta_j,\ S \text{ is minimal} \Bigg\}$$

In other words, $\mathcal{C}^e_j$ contains all subsets of $\delta^-(j)$ that are a minimal activation set for some value of the random data (i.e., θ and φ) within the uncertainty set. By definition, $\mathcal{C}_j \subseteq \mathcal{C}^e_j$. Moreover, it is easy to check that $\mathcal{C}^e_j$ coincides with $\mathcal{C}_j$ whenever $\Delta_N = 0$ and $\Delta_A = 0$. An algorithm to compute $\mathcal{C}^e_j$ is given in Algorithm 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 $\mathrm{RI}_{x,\theta=\bar{\theta},\varphi=\bar{\varphi}}(y)$ 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 Sect. 5.1.

Note that the number of sets in $\mathcal{C}^e_j$ 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 assumption that all the sets in

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

```
1: if first_to_add > $|\delta^-(j)|$ then return false
2: any_generated ← 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 ← 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 ← first_to_add to $|\delta^-(j)|$ do
8:     next_generated ← 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 ← true
   return any_generated
```

Algorithm 2: Function RecGenExt(j, S, first_to_add, $\mathcal{C}_j^e$). We assume w.l.o.g. (up to relabeling) that $\delta^-(j) := \{1, \ldots, |\delta^-(j)|\}$, 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, $S \leftarrow \emptyset$, 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 in $\mathcal{C}_j^e$ are known explicitly. To do so, we introduce some additional notation.

Definition 4 For every $\bar{y} \in \{0,1\}^n$, let $\theta^{\bar{y}}, \varphi^{\bar{y}}$ be the optimal values of θ, φ for problem $\mathrm{RI}_{x,\theta,\varphi}(\bar{y})$. We define the corresponding *collection of seed-dependent minimal activation sets* $\mathcal{C}_j^{\bar{y}}$ as:

$$\mathcal{C}_j^{\bar{y}} = \left\{ S \subseteq \delta^-(j) : \sum_{i \in S} (w_{ij} - \varphi_{ij}^{\bar{y}}) \geq t_j + \theta_j^{\bar{y}}, S \text{ is minimal} \right\}.$$

In other words, $\mathcal{C}_j^{\bar{y}}$ is the collection of activation sets that are minimal for a given $\theta^{\bar{y}}, \varphi^{\bar{y}}$. Notice that $\mathcal{C}_j^{\bar{y}} \subseteq \mathcal{C}_j^e$ for all $\bar{y}$.

We show that (R-IMP) can be solved via the solution of an exponential number of problems of the form (DUAL-θ0-φ0), with different objective functions. Furthermore, this can in principle be solved using just one Branch-and-Bound tree. Consider the following problem, where $\bar{y} \in \{0,1\}^n$.

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

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 $RI_{x,\theta,\varphi}(\tilde{y})$ [Eq. (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:

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

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

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

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

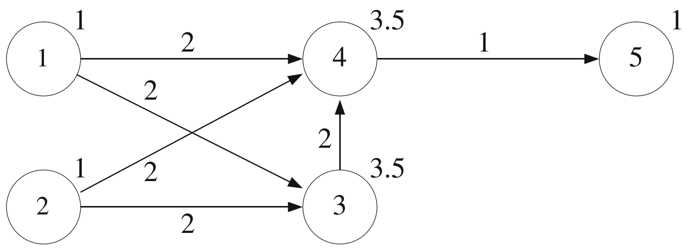


Fig. 2 Graph discussed in the example related to Theorem 4. Node labels are indicated inside each node, node thresholds are indicated above and to the right of each node, and arc weights are indicated next to each arc

Theorem (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 $\mathcal{C}^e_j$ of activation sets for all $j \in V$, using Algorithm 2. It is easy to see that for a given $\bar{y}$, each activation set $S \in \mathcal{C}^e_j$ is valid in the range $[t_j, \min\{(1+\Delta_N)t_j, \sum_{i\in S}(w_{ij} - \varphi^{\bar{y}}_{ij})\}]$ and it is minimal if the threshold is strictly greater than $\sum_{i\in S}(w_{ij} - \varphi^{\bar{y}}_{ij}) - \min_{i\in S}\{w_{ij} - \varphi^{\bar{y}}_{ij}\}$. 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.)

Correctness of this algorithm follows by Theorem 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 θ, φ) 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 θ, φ 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 Sect. 6.

As every distinct $\bar{y}$ yields a different first row in (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 Theorem 4, in the sense that the linear programming relaxation to the problem would no longer yield valid dual bounds.[1] Consider the graph depicted in Fig. 2. We assume that $B_N = 0$, $B_A = 1$, $q = 2$. The extended collections of minimal activation sets for the three nodes that can be activated by other nodes are:

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

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 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 $\mathrm{RI}_{x,\theta,\varphi}(\bar{y})$. In our example, we assume that the optimal solutions have the following values of φ (θ_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 φ_{23} disables activation set S_1. Problem (R-IMP-$\bar{y}$) with all the constraints corresponding to the three $\bar{y}$ indicated above reads:

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

[1] 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 Theorem 4 of the IPCO version: it is fixed in this paper. Experiments show that ≈ 25% of the solutions found in the IPCO paper are suboptimal.

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:

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

In the dual (7), γ 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 γ variables. In contrast, the r.h.s. vector of (5) is fixed and corresponds to having a single nonzero γ variable equal to 1. Because of this, the value of (7) can be lower than $\mathrm{RI}_{x,\theta,\varphi}(\bar{y})$, yielding an invalid bound, see Theorem 4. In this example, (7) has optimal objective function value 3.5, and the optimal variables are (we report only nonzeroes):

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

This is not a valid upper bound, because $\mathrm{RI}_{x,\theta,\varphi}(11000)$ 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, $\mathcal{C}^e_j$ 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:

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

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:

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

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

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

Our aim is to determine $S \in \mathcal{C}_j^e$, 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 $S \in C_j^e \backslash C_j^{\bar{y}}$; we use a binary variable β 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 $\bar{w}_j = \max_{i \in \delta^-(j)} w_{ij}$:

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

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 $j \in V$. We note that, from a computational point of view, no-good cuts tend to be weak in practice, and their main purpose in our scheme is to avoid cycling, rather than improving the strength of the formulation for the pricing problem. 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 Sect. 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 in $\mathcal{C}_j^e$ are computed in a preprocessing step. We use this implementation on graphs of smaller size; it is called Exact in Sect. 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 $\mathrm{RI}_{x,\theta,\varphi}(\tilde{y})$.
- If $\mathrm{RI}_{x,\theta,\varphi}(\tilde{y}) > L$, save $\tilde{y}$, update $L \leftarrow \mathrm{RI}_{x,\theta,\varphi}(\tilde{y})$, 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 $\mathrm{RI}_{x,\theta,\varphi}(\tilde{y}) < L$, reject the incumbent and continue.
- Else, if the value of the current incumbent matches the value of $\mathrm{RI}_{x,\theta,\varphi}(\tilde{y})$, then accept the incumbent and continue; otherwise, reject the incumbent and continue.

Correctness of this algorithm is a direct consequence of the discussion in Sect. 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 $\mathrm{RI}_{x,\theta,\varphi}(\tilde{y}) > L$, 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 we use on large graphs; this is called HEURCG in Sect. 7. The algorithm uses EXACT as a subroutine. It requires parameters `nic` (number of initial columns), `mpi` (maximum number of "priced" nodes per iteration), `mci` (maximum number of column generation subproblems per iteration), `mcr` (maximum number of columns added per round), and it works as follows.

- Generate an initial set of columns C, with $|C| = \texttt{nic}$. To generate the initial set of columns we apply RECGENEXT 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 `nic` columns, keeping all columns generated this way (which may therefore be more than `nic`).
- 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 $\texttt{mpi} \times |V|$ randomly chosen nodes $j \in V$. The pricing step performs a pricing iteration until: (i) the bound of the LP matches $\mathrm{RI}_{x,\theta,\varphi}(\tilde{y})$, or (ii) we have performed $\texttt{mci} \times |V|$ pricing iterations, or (iii) the number of generated columns per pricing step is greater than $\texttt{mcr} \times |V|$.
  - Let C' be the newly generated columns. Set $C \leftarrow C \cup C'$.

In our implementation we use the following values: $\texttt{nic} = 2500$, $\texttt{mpi} = 0.025$, $\texttt{mci} = 2$ and $\texttt{mcr} = 2$. This choice is motivated by the empirical analysis discussed in "Appendix C".

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. The code is available via GitHub, see [24]. We activate the numerical emphasis setting because of some difficulties faced during our numerical evaluation (on some graphs, model (DUAL-$\theta 0$-$\varphi 0$) 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 (2.00 GHz, 64 GB RAM).

We consider graphs belonging to the class *SW* ("small world"), described in Fischetti et al. [12], Watts and Strogatz [25]. 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. These graph as well as some additional instances are freely available via GitHub at https://github.com/sartorg/robinmax [24]. We also tested a directed version of the Erdos-Rényi random graph Erdos and Rényi [11]: the conclusions of the study are similar and are therefore not reported.

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 Δ_N, Δ_A are always equal to 0.1.

Tables 1, 2 and 3 illustrate the results obtained on *Small Graphs* with Exact, where all extended collections of activation sets are generated in a preprocessing phase with Algorithm 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 Fischetti et al. [12]. 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 $(S \backslash \{u\}) \cup \{v\}$, where $u \in S$ and $v \in V \backslash S$, 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

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

			Time (s)			opt (frac)			gap (%)			Activation sets (#)		
		B_A	0	1	2	0	1	2	0	1	2	0	1	2
k	s	B_N												
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
	5	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
	8	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
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
	5	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
	8	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

Time limit 3600 s

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

		B_A	Time (s)			opt (frac)			gap (%)			Activation sets (#)		
			0	1	2	0	1	2	0	1	2	0	1	2
k	s	B_N												
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	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	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	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	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

Time limit 3600 s

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

		B$_A$	Time (s)			opt (frac)			gap (%)			Activation sets (#)		
k	s	B$_N$	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	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	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	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	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

Time limit 3600 s

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

			B_N	B_A	0		25		50	
n	k	Seeds			HEURCG	2opt	HEURCG	2opt	HEURCG	2opt
2000	12	200	0		240, 3	201, 7	209, 4	200, 4	209, 3	200, 5
			25		211, 3	200, 7	204, 8	200, 1	204, 3	200, 1
			50		210, 8	200, 6	204, 6	200, 2	204, 0	200, 1
		300	0		230, 1	301, 6	210, 1	300, 9	210, 2	301, 1
			25		211, 7	300, 9	204, 2	300, 8	202, 8	300, 5
			50		211, 6	301, 1	203, 5	300, 3	202, 8	300, 5
	16	200	0		374, 4	200, 5	326, 5	200, 1	327, 0	200
			25		327, 8	200, 1	315, 5	200, 1	309, 7	200
			50		326, 4	200, 1	309, 9	200, 1	308, 2	200
		300	0		357, 9	301, 2	328, 4	300, 4	323, 3	300, 4
			25		328, 8	300, 7	314, 4	300, 3	310, 0	300, 3
			50		327, 4	300, 6	310, 9	300, 2	308, 8	300, 1
5000	12	500	0		588, 3	501, 7	537, 4	500, 6	528, 9	500, 6
			25		539, 3	500, 5	523, 2	500, 6	515, 7	500, 4
			50		531, 0	500, 6	512, 9	500, 6	512, 3	500, 5
		750	0		890, 2	753, 8	826, 6	751, 2	808, 2	750, 9
			25		825, 9	751, 2	808, 6	750, 7	793, 0	750, 9
			50		815, 0	751, 2	795, 2	750, 8	790, 6	750, 8
	16	500	0		561, 8	500, 2	531, 8	500	527, 6	500
			25		534, 9	500, 1	526, 4	500	518, 5	500
			50		534, 4	500, 1	517, 9	500	513, 7	500
		750	0		847, 4	750, 8	818, 8	750	805, 3	750
			25		814, 3	750, 3	801, 6	750	790, 4	750
			50		813, 6	750, 3	795, 3	750	787, 1	750

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.

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.

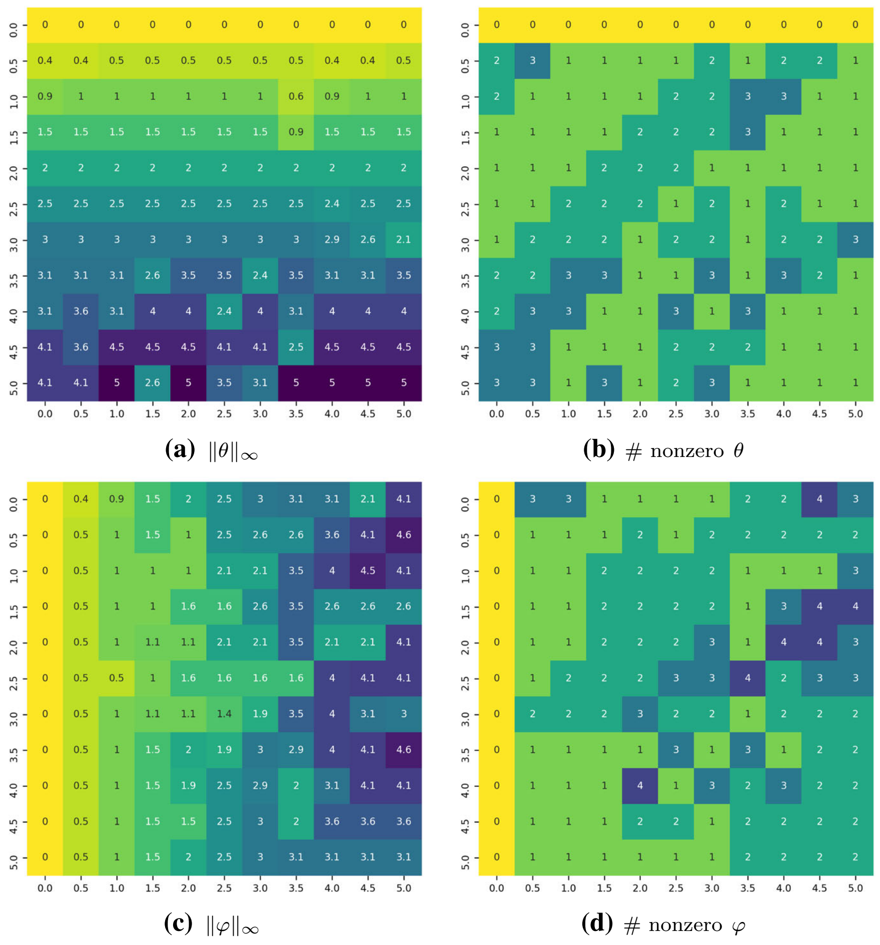


Fig. 3 The maximum values of θ and φ are reported in (**a**) and (**c**). The number of nonzeros of θ and φ are reported in (**b**) and (**d**). In each subfigure, the value of B_N increases from top to bottom and the value of B_A increases from left to right

7.1 The price of robustness: a case study

Our results in Sect. 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

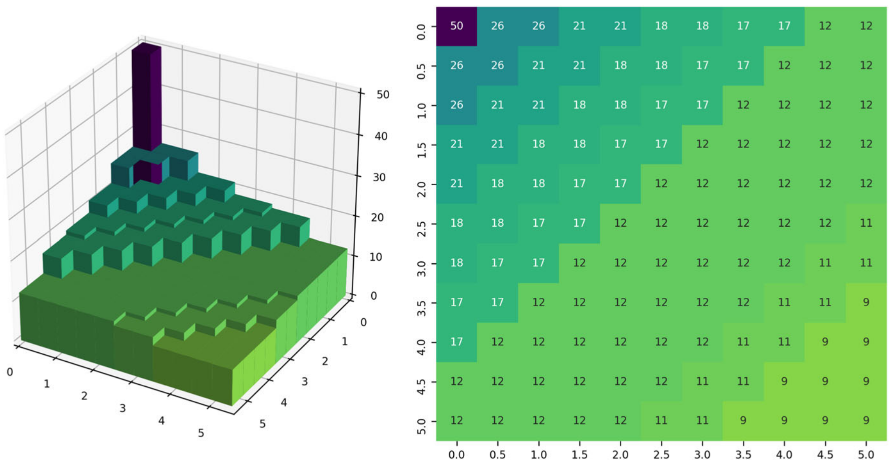


Fig. 4 Optimal value of the problem for different values of B_N and B_A

possible combinations. We set $\varDelta_A = 1$ and $\varDelta_N = \frac{B_N}{\min_j t_j}$, so that no upper bounds on the values of θ and φ are imposed.

In Fig. 3 we report the maximum value of the θ and φ variables, i.e., the ℓ_∞-norm of the corresponding vector, as well as the number of nonzero θ and φ variables at the optimum. Each subtable shows the values of $\|\theta\|_\infty$, $\|\varphi\|_\infty$, # nonzero θ and # nonzero φ 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 θ and φ, 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.

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.

Acknowledgements Part of G. Nannicini's research was conducted with support by the Simons Foundation and by the Mathematisches Forschungsinstitut Oberwolfach.

A Table of notation

$\delta^-(j), \delta^+(j)$	instar and outstar of node j
$y \in \{0, 1\}^n$	incidence vector of the seeds
$x \in \{0, 1\}^n$	incidence vector of the activated nodes
t_i	threshold of node i
w_{ij}	weight of arc $\{i, j\}$
Δ_N	maximal node threshold variation (expressed as percentage of t)
B_N	total budget of threshold variations
Δ_A	maximal arc weight variation (expressed as percentage of w)
B_A	total budget of weight variations
P	set of robustness parameters $\{B_N, \Delta_N, B_A, \Delta_A\}$
θ_j	increase of node threshold t_j in a robust solution
φ_{ij}	decrease of weight w_{ij} in a robust solution
$\mathrm{RI}_{x,\theta,\varphi}(\bar{y})$	total amount of influence that spreads on the graph for the given set of seeds $\bar{y}$, formulated as an optimization problem with decision variables x, θ and φ
$\mathrm{RI}^P_{x,\theta,\varphi}(\bar{y})$	problem $\mathrm{RI}_{x,\theta,\varphi}(\bar{y})$ for a given set of robustness parameters P
(R-IMP)	mathematical model for the robust IMP with q activation seeds, formulated as bilevel optimization problem
$\mathcal{C}_j$	collection of minimal activation sets
$\mathrm{AS}_x(y)$	total amount of influence that spreads on the graph for the given set of seeds $\bar{y}$ with $\theta = \varphi = 0$, computed with a formulation based on activation sets
π, μ	variables of the dual of $\mathrm{AS}_x(\bar{y})$, for a fixed $\bar{y}$
(IMP-θ0-φ0)	mathematical model for IMP, formulated using $\mathrm{AS}_x(y)$
(DUAL-θ0-φ0)	mathematical model for IMP, formulated using the dual of $\mathrm{AS}_x(y)$
$\mathcal{C}^e_j$	extended collection of minimal activation sets
$\mathcal{C}^{\bar{y}}_j$	collection of seed-dependent minimal activation sets
(R-IMP-$\bar{y}$)	mathematical model used to obtain valid dual bounds for IMP, parametric in $\bar{y}$
(PRICE-j)	mathematical model used as pricing to generate varialbes associated to minimal active sets with a negative reduced cost, one for each node j
ψ, β, α	variables of (PRICE-j)

B Big-M discussion

To find a value of big-M that ensures a valid formulation for the problem obtained by dualizing the inner problem AS in IMP-θ0-φ0, we can amend the dual formulation putting an upper bound on the dual variables: μ_j:

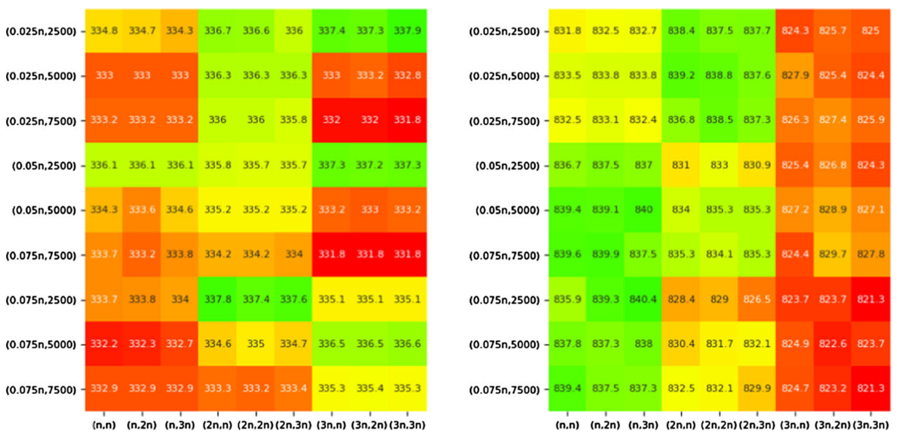
$$\left.\begin{array}{rrr} \max_{\pi,\mu} & \sum_{j \in V} \sum_{S \in \mathcal{C}_j} (|S|-1)\pi_{j,S} + \sum_{j \in V} \mu_j \bar{y}_j & \\ \forall j \in V & \sum_{k \in \delta^+(j)} \sum_{S \in \mathcal{C}_k : j \in S} \pi_{k,S} - \sum_{S \in \mathcal{C}_j} \pi_{j,S} + \mu_j & \leq 1 \\ \forall j \in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

We need to find a value of U that does not change the solution to this problem. If we go back to the primal, we obtain:

$$\left.\begin{array}{rrr} & \min \sum_{j \in V} x_j + \sum_{j \in V} U u_j & \\ \forall j \in V, \forall S \in \mathcal{C}_j & \sum_{i \in S} x_i - x_j & \leq |S|-1 \\ \forall j \in V & x_j + u_j & \geq \bar{y}_j \\ \forall j \in V & u_j & \geq 0, \end{array}\right\}$$

where u_j is the variable associated with the dual constraints $\mu_j \leq U$. We can provide examples of graphs where $U = n$ does not suffice. Suppose we have a directed graph with n nodes: a triangle $1 \rightarrow 2, 2 \rightarrow 3, 3 \rightarrow 1$, three edges $1 \rightarrow 4, 2 \rightarrow 4, 3 \rightarrow 4$, and finally a chain $4 \rightarrow 5, 5 \rightarrow 6, \ldots, n-1 \rightarrow n$. Suppose all edge weights are 1 and all activation thresholds are 1, except for node 4 that has activation threshold equal to 3. The initial node seed is node 1, i.e., we have $\bar{y}_1 = 1$, $\bar{y}_k = 0$ for $k \geq 2$. In this case, if U is set correctly, i.e., large enough that all u_j variables are 0, the solution to the primal should have $x_k = 1$ for all k: all nodes are active, for an objective function value of n. This is because node 1 activates node 2, which activates node 3, these three nodes activate node 4, and then the entire chain activates. However after we introduce variables u_j in the primal, we can set $x_1 = \frac{2}{3}, u_1 = \frac{1}{3}, x_2 = \frac{2}{3}, x_3 = \frac{2}{3}$: now the sum $x_1 + x_2 + x_3 = 2$ is not enough to activate node 4 (associated with the constraint $x_1 + x_2 + x_3 - x_4 \leq 2$), so that the total objective function value is $x_1 + x_2 + x_3 + U u_1 = 2 + \frac{1}{3}U < n$ as long as $U < 3(n-2)$. Thus, this shows that we need $U > n$. In fact the example can be easily extended (using a cycle of length d, rather than a triangle) to show that we need at least $U \geq nd$, where d is the maximum indegree of a node in the graph. Numerically, we found counterexamples where even $U = n^2$ did not suffice; these examples were difficult to study analytically.

Given these difficulties, a straightforward big-M formulation is likely to fail. On the other hand, the indicator constraint version gives more freedom to CPLEX in tightening the big-M or switching to alternative formulation. In preliminary experiments we found that the use of indicator constraints leads to a numerically more stable implementation.



(a) n = 2000 **(b)** n = 5000

Fig. 5 In each subfigure, the value of (`mpi`,`nic`) are reported on the vertical axis, while the values of (`mci`,`mcr`) are reported on the horizonal axis

C Analysis of the parameters used in the heuristic comparison

To choose the parameters to be used in algorithm *HeurCG* we tested the following combinations of values: `nic` $\in \{2500, 5000, 7500\}$ `mpi` $\in \{0.025, 0.05, 0.75\}$, `mci` $\in \{1, 2, 3\}$ and `mcr` $\in \{1, 2, 3\}$, for a total of 81 different configurations. Each configuration has been used to solve a subset of *Large graphs* with 2000 and 5000 nodes, an average node degree $k = 12$, a rewiring probability $b \in \{0.1, 0.3\}$, $0.15n$ starting seeds and 5 random instances for each combination of settings. The robustness parameters chosen are $\mathrm{B}_A \in \{0, 25\}$ and $\mathrm{B}_N \in \{0, 25\}$.

In Fig. 5 we show the optimal value obtained, where each entry corresponds to the average computed over 40 instances. We can see that the variance of the optimal value is limited: for the instances with 2000 nodes the maximum difference between the best and the worse average is equal to 6.1, while for the instances with 5000 nodes, it is equal to 19.1. The value used for `mci` has a small impact on the performance of the algorithm, therefore we decided to use the intermediate value of 2. The parameter that seems to have the largest impact is `mcr`. These experiments suggest setting `mcr` $= 2$: even if in some cases the best performance is achieved with values of `mcr` different from 2, the selected setting is the one providing the most consistent results across the graph sizes considered. Finally, for what it concerns the other parameters, we selected `nic` $= 2500$ and `mpi` $= 0.025$ because they show the best performance when `mcr` is equal to 2.

References

1. Ackerman, E., Ben-Zwi, O., Wolfovitz, G.: Combinatorial model and bounds for target set selection. Theor. Comput. Sci. **411**(44–46), 4017–4022 (2010)

2. Banerjee, S., Jenamani, M., Pratihar, D.K.: A survey on influence maximization in a social network. arXiv preprint arXiv:1808.05502 (2018)
3. Belotti, P., Bonami, P., Fischetti, M., Lodi, A., Monaci, M., Nogales-Gómez, A., Salvagnin, D.: On handling indicator constraints in mixed integer programming. Comput. Optim. Appl. **65**(3), 545–566 (2016)
4. Ben-Tal, A., Nemirovski, A.: Selected topics in robust convex optimization. Math. Program. **112**(1), 125–158 (2008)
5. Bertsimas, D., Sim, M.: The price of robustness. Oper. Res. **52**(1), 35–53 (2004)
6. Bertsimas, D., Brown, D.B., Caramanis, C.: Theory and applications of robust optimization. SIAM Rev. **53**(3), 464–501 (2011)
7. Buchheim, C., Kurtz, J.: Robust combinatorial optimization under convex and discrete cost uncertainty. EURO J. Comput. Optim. **6**(3), 211–238 (2018)
8. 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)
9. Domingos, P., Richardson, M.: Mining the network value of customers. In: Hammer, P.L., Johnson E.L., Korte, B.H., Nemhauser, G.L. (eds.) Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 57–66 (2001)
10. 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)
11. Erdos, P., Rényi, A.: On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci **5**(1), 17–60 (1960)
12. Fischetti, M., Kahr, M., Leitner, M., Monaci, M., Ruthmair, M.: Least cost influence propagation in (social) networks. Math. Program. **170**(1), 293–325 (2018). (ISSN: 1436-4646)
13. 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)
14. Gunnec, D.: Integrating social network effects in product design and diffusion. Ph.D. thesis, XXX (2012)
15. He, X., Kempe, D.: Stability and robustness in influence maximization. ACM Trans. Knowl. Discov. Data (TKDD) **12**(6), 66 (2018)
16. Kempe, D., Kleinberg, J., Tardos, É.: Maximizing the spread of influence through a social network. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '03, pp. 137–146, New York, NY, USA. ACM. ISBN: 1-58113-737-0 (2003)
17. Kempe, D., Kleinberg, J., Tardos, É.: Maximizing the spread of influence through a social network. Theory Comput. **11**(4), 105–147 (2015)
18. Könemann, J., Sadeghabad, 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)
19. Li, Y., Fan, J., Wang, Y., Tan, K.-L.: Influence maximization on social graphs: a survey. IEEE Trans. Knowl. Data Eng. **30**(10), 1852–1872 (2018)
20. 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. Ind. Eng. Chem. Res. **50**(18), 10567–10603 (2011)
21. McCormick, G.P.: Computability of global solutions to factorable nonconvex programs: part I—convex underestimating problems. Math. Program. **10**(1), 147–175 (1976)
22. 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)
23. 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)
24. Sartor, G., Traversi, E., Calvo, R.W.: RobInMax: An open-source library for robust influence maximization (2020). https://doi.org/10.5281/zenodo.3692697
25. Watts, D.J., Strogatz, S.H.: Collective dynamics of "small-world" networks. Nature **393**(6684), 440 (1998)

26. Wu, H.-H., Küçükyavuz, S.: A two-stage stochastic programming approach for influence maximization in social networks. Comput. Optim. Appl. **69**(3), 563–595 (2018)