Least cost influence propagation in (social) networks*

Matteo Fischetti[1], Michael Kahr[2], Markus Leitner[2], Michele Monaci[3] and Mario Ruthmair[2]

[1]DEI, University of Padua, Italy. matteo.fischetti@unipd.it
[2]ISOR, University of Vienna, Austria.
{m.kahr|markus.leitner|mario.ruthmair}@univie.ac.at
[3]DEI, University of Bologna, Italy. michele.monaci@unibo.it

January 30, 2018

Abstract

Influence maximization problems aim to identify key players in (social) networks and are typically motivated from viral marketing. In this work, we introduce and study the Generalized Least Cost Influence Problem (GLCIP) that generalizes many previously considered problem variants and allows to overcome some of their limitations. A formulation that is based on the concept of activation functions is proposed together with strengthening inequalities. Exact and heuristic solution methods are developed and compared for the new problem. Our computational results also show that our approaches outperform the state-of-the-art on relevant, special cases of the GLCIP.

Keywords: Influence maximization · Mixed-integer programming · Social network analysis

1 Introduction

An increased interest in studying and solving optimization problems related to the propagation of influence in social networks can be observed recently; see, e.g, [5, 16] and the references therein. Many of these problems are concerned with the identification of key players in a social network that are crucial for the process of influence propagation. Consider, for instance, a company that performs a promotion strategy for a product on a social network. A somewhat natural assumption thereby is that customers which could be convinced about the product will exert influence on their connections, thus, increasing the probability that the latter will be convinced as well. In that setting, a crucial point is to decide which users shall be convinced initially (e.g., via discounts, free products, or monetary incentives) that will trigger the propagation process. Typical goals are either to convince a certain fraction of all users with minimum budget, or to maximize the number of convinced users while respecting a given budget constraint. Besides (viral) marketing, similar influence propagation models have applications in, e.g., Epidemiology [7]. A common technique to model the propagation process is based on the *linear threshold model* initially proposed by Granovetter [9], in which individuals in a social network get *active* (e.g., adopt an information or a

*Supported by the Vienna Science and Technology Fund through project ICT15-014 and MiUR, Italy (PRIN 2015 project).

product) if the sum of influences received from their active neighbors reaches a predefined threshold, i.e., its *hurdle*. Alternatively, individuals can be targeted to act as seeds of the propagation process in which case they are activated initially. Recently, this concept has been generalized through the consideration of partial *incentives* (e.g., discounts) given to individuals that allow to reduce their hurdle [12]. Hence, individuals can also get active by a combination of influence received from their neighbors and incentives.

Overview and main contributions. In this article, we introduce and study a new optimization problem in the domain of influence propagation to which we refer to as the *Generalized Least Cost Influence Problem (GLCIP)*. As we will show in Section 2, the GLCIP contains previously considered problems as special cases and provides a unifying framework for several problems that have been studied separately so far. Besides, our new problem allows to overcome certain limitations of previous models that might prohibit their application in real world. In particular, we introduce the concept of activation functions which are used to decide whether an individual gets active or not, see Definition 1. These activation functions generalize previously considered so-called threshold functions by incorporating partial node incentives. In Section 3, we propose a novel Integer Linear Programming (ILP) formulation for the GLCIP as well as strengthening inequalities. An alternative ILP formulation that generalizes previously proposed formulations for a particular class of activation functions is proposed in Section 4. Details of the developed solution methods based on row and column generation are given in Section 5. We also show that our new formulation enables the consideration of arbitrary (e.g., non-linear) activation functions. Finally, in Section 6 we perform an extensive computational study also including previously considered special cases of the GLCIP, and show that our approach outperforms existing methods.

Problem Definition. The GLCIP is defined on a directed graph $G = (V, A)$ whose node set V represents the individuals of a considered (social) network, and whose arc set A models established relations between them. The strength of *influence* a node $i \in V$ exerts on $j \in V$ is indicated by value $d_{ij} > 0$, which is associated to each arc $(i, j) \in A$. *Hurdles* $h_i > 0$ define the thresholds that need to be reached through neighboring influence and incentives in order to *activate* node $i \in V$. Potential *incentives* are defined by set P_i together with costs $w_{ip} \geq 0$ for offering incentive $p \in P_i$ to node $i \in V$. Parameter α, $0 \leq \alpha \leq 1$, defines the minimum fraction of nodes that need to be activated. A feasible solution $\mathcal{S} = (V^{\mathcal{S}}, N^{\mathcal{S}}, p^{\mathcal{S}})$ to the GLCIP consists of a set of activated nodes $V^{\mathcal{S}} \subset V$ such that $|V^{\mathcal{S}}| \geq \lceil \alpha |V| \rceil$, a set of influencing neighbors $N_i^{\mathcal{S}} \subseteq N_i = \{j \in V \mid (j, i) \in A\}$ and an incentive $p_i^{\mathcal{S}} \in P_i$ (possibly equal to zero) for each active node $i \in V^{\mathcal{S}}$ such that i can be activated by node set $N_i^{\mathcal{S}}$ together with incentive $p_i^{\mathcal{S}}$. Let $U \subseteq N_i$ be the set of active neighbors of node i, $p \in P_i$ is the incentive given to it, and $f_i : 2^{N_i} \times P_i \to \mathbb{R}_+$ is its *activation function*; see, Definition 1 for a formal definition. Then, a previously inactive node $i \in V$ gets active in the current step of the propagation process if and only if $f_i(U, p) \geq h_i$, i.e., if the influence received from its neighbors and a potential incentive reaches its hurdle. The objective of the GLCIP is to find a solution that minimizes the total costs $\sum_{i \in V^{\mathcal{S}}} w_{ip_i^{\mathcal{S}}}$ for paid incentives.

Observe that above activation functions are a natural extension of *threshold functions* $g_i : 2^{N_i} \to \mathbb{R}_+$ that are extensively discussed by Kempe et al. [16]. These threshold functions are *monotone*, map the empty set to zero, and consider the subset of previously activated neighbors only. Note that Kempe et al. [16] focus on the case in which all individual influences as well as each node's hurdle are between zero and one, i.e., $0 \leq d_{ij} \leq 1$ for all $(i, j) \in A$, and $0 \leq h_i \leq 1$ for all

$i \in V$, and for which $0 \leq g_i(U) \leq 1$ holds for each $U \subseteq N_i$. Based on these assumptions, they point out two particularly relevant cases of a general threshold model, namely, the linear and the submodular threshold model. While in the linear case the total influence of subset U is simply defined by the sum of individual influences, i.e., $g_i(U) = \min(1, \sum_{j \in U} d_{ji})$, the submodular one allows to account for diminishing influence of additional nodes. Formally, for each node $j \in N_i$ and two subsets $U \subseteq U' \subseteq N_i \setminus \{j\}$, the associated submodular threshold function for node $i \in V$ must satisfy $g_i(U \cup \{j\}) - g_i(U) \geq g_i(U' \cup \{j\}) - g_i(U')$. While both models have been used to derive approximation results (see, Kempe et al. [14, 16]), only the linear threshold model has received significant attention from a computational viewpoint; see, e.g., [4, 12, 13] and Section 2 for more details.

Notation and Assumptions. Besides using $N_i = \{j \in V \mid (j,i) \in A\}$ to denote the node set that may directly influence node $i \in V$, we denote its power set by $\mathcal{N}_i = 2^{N_i}$. Without loss of generality (see Proposition 1), a strict total ordering is defined on the set of potential incentives P_i for each node $i \in V$. To simplify notation, each P_i includes a special element 0 (i.e., no incentive is given and no cost is incurred), and all the remaining incentives $p \in P_i \setminus \{0\}$ satisfy $p > 0$ and $w_{ip} > 0$. There exists at least one element $p \in P_i$ that can activate node $i \in V$ without additional influence from neighboring nodes, i.e., such that $f_i(\emptyset, p) \geq h_i$ holds. If an instance does not contain such an incentive for some node i, we add an artificial incentive $p = \infty$ with $w_{ip} = \infty$ without changing the optimal solution value (if the resulting optimal costs are equal to infinity, the original instance is infeasible). We also observe that it is sufficient to consider at most one incentive such that $f_i(\emptyset, p) \geq h_i$ for each node i. In a preprocessing step, we first compute the cheapest incentive p that is sufficient to activate a node without receiving influence from its neighbors, i.e., $p = \operatorname{argmin}_{p' \in P_i}\{w_{ip'} \mid f_i(\emptyset, p') \geq h_i\}$, and then remove all possibly existing incentives $p'' \neq p$ with $w_{ip''} \geq w_{ip}$. Finally, we assume that an active node remains active throughout the propagation process and that, for a given node $i \in V$, set $U \in N_i$ and incentive $p \in P_i$, the propagation function $f_i(U, p)$ can be evaluated in polynomial time.

Definition 1 formally defines the concept of an activation function, while particular classes of such functions that will be relevant for the remainder of this article are introduced in Definitions 2 and 3.

Definition 1 (Activation Function)**.** *A monotone function $f_i : \mathcal{N}_i \times P_i \to \mathbb{R}_+$ defined for $i \in V$ is an* activation function *if it satisfies $f_i(\emptyset, 0) = 0$. Function f_i is monotone if and only if $U \subset U' \in \mathcal{N}_i$ implies that $f_i(U, p) \leq f_i(U', p)$, $\forall p \in P_i$, and $p \leq p' \in P_i$ implies that $f_i(U, p) \leq f_i(U, p')$, $\forall U \in \mathcal{N}_i$.*

Definition 2 (Influence-Monotone Activation Function)**.** *An activation function $f_i : \mathcal{N}_i \times P_i \to \mathbb{R}_+$ of node $i \in V$ is called* influence-monotone *if and only if $f_i(U \cup \{j\}, p) \geq f_i(U \cup \{k\}, p)$ holds for each incentive $p \in P_i$, each set $U \in \mathcal{N}_i$, and every pair of nodes $j \neq k \in N_i \setminus U$ such that $d_{ji} \geq d_{ki}$.*

Definition 3 (Additively Separable Activation Function)**.** *An activation function $f_i : \mathcal{N}_i \times P_i \to \mathbb{R}_+$ of node $i \in V$ is called* additively separable *iff there exists a threshold function $g_i : \mathcal{N}_i \to \mathbb{R}_+$ such that $f_i(U, p) = g_i(U) + p$ holds for all $U \in \mathcal{N}_i$ and $p \in P_i$. An additively separable activation function is called* additively separable and linear *iff its embedded threshold function is linear w.r.t. to the influence values d_{ji} of neighbors $j \in N_i$.*

Proposition 1 shows that the strict ordering of incentives together with the required monotonicity of activation functions allows us to assume without loss of generality that $p < p'$ implies $w_{ip} < w_{ip'}$.

Proposition 1. *Let $f_i : \mathcal{N}_i \times P_i \to \mathbb{R}_+$ be an activation function of node $i \in V$. Then, we may assume without loss of generality that higher incentives induce higher costs, i.e., $w_{ip} < w_{ip'}$ holds for each $p, p' \in P_i$ such that $p < p'$.*

Proof. Let $p, p' \in P_i$ be two incentives that do not satisfy the condition of the proposition, i.e., $p < p'$ and $w_{ip} \geq w_{ip'}$ and let $\mathcal{S}$ be an arbitrary solution giving incentive $p_i^{\mathcal{S}} = p$ to node i. Since f_i is monotone, a solution $\mathcal{S}'$ in which p is replaced by p' activates at least the same number of nodes, and is at most as expensive as $\mathcal{S}$ by assumption. Thus, if $\mathcal{S}$ is feasible then $\mathcal{S}'$ is feasible as well, and its objective value is at most the one of $\mathcal{S}$. Consequently, incentive p can be removed from P_i. Repeating this procedure for every pair of incentives that does not satisfy the conditions of the proposition, the result follows. □

Several steps of our solution method will require the computation of the cheapest incentive that activates a particular node for a fixed set of influencing (active) neighbors. Proposition 2 states that this can be performed efficiently if the associated activation function can be evaluated efficiently. The validity of the statement follows from exploiting the required strict total ordering of the elements of P_i and by applying, e.g., binary search.

Proposition 2. *Consider a node $i \in V$ and an arbitrary activation function $f_i : \mathcal{N}_i \times P_i \to \mathbb{R}_+$. For each $U \subseteq N_i$, the cheapest incentive $p_i(U) \in P_i$ required to activate node i if all nodes from U are active, i.e., $p_i(U) = argmin_{p \in P_i}\{w_{ip} \mid f_i(U, p) \geq h_i\}$, can be computed in at most $\mathcal{O}(\log(|P_i|))$ steps. Thereby, each step requires the evalution of function f_i.*

Definition 4 formally introduces the concept of propagation graphs.

Definition 4 (Induced Propagation Graph)**.** *Let $\mathcal{S} = (V^{\mathcal{S}}, N^{\mathcal{S}}, p^{\mathcal{S}})$ be a solution to the GLCIP. Subgraph $G' = (V^{\mathcal{S}}, A')$ of G is called the* propagation graph induced by $\mathcal{S}$ *if and only if G' satisfies the following conditions: (i) $A' = \{(j, i) \in A \mid \{i, j\} \subseteq V^{\mathcal{S}}, j \in N_i^{\mathcal{S}}\}$ includes all arcs on which influence is exerted, and (ii) G' is acyclic, i.e., contains no directed cycle.*

The last condition of Definition 4 follows from the temporal aspect of the activation process and is important to forbid activating nodes by cycles that are not triggered through incentives. We also observe that (symmetric) solutions may exist in which offered incentives coincide (thus having identical objective values), but which differ with respect to the induced propagation graph. Besides such symmetric solutions that differ with respect to active nodes, there also exist solutions in which only the influencing sets (and thus the arc sets of their respective induced propagation graphs) differ. As detailed in Theorem 1, one can, however, find an induced propagation graph with maximal set $V^{\mathcal{S}}$ in polynomial time in case all offered incentives are fixed.

Theorem 1. *Let $p_i \in P_i, \forall i \in V$, be the incentives associated to each node in an GLCIP instance. Then, a solution $\mathcal{S} = (V^{\mathcal{S}}, N^{\mathcal{S}}, p)$ with maximal set $V^{\mathcal{S}}$ can be computed in polynomial time.*

Proof. Algorithm 1 computes a solution with maximal set V^{S} in polynomial time. It starts with an empty solution and triggers propagation from all seed nodes activated solely by incentives exerting influence to all not yet activated neighbors. If one of these neighbors gets activated, propagation continues until no more nodes can be activated. At each iteration, we activate all nodes i for which the activation function f_i is not less than hurdle h_i. Since activating a node cannot prevent another node from becoming activated, the final set V^S is maximal. As to the time complexity, we observe that, for each arc $(i, j) \in A$ we evaluate function f_j at most once, when node i is activated

Algorithm 1: Computation of solution with maximal node set for fixed incentives $p_i, \forall i \in V$.

$V^{\mathcal{S}} = \emptyset, N_i^{\mathcal{S}} = \emptyset, \forall i \in V$ // start with empty solution
$Q = \{i \in V : f_i(\emptyset, p_i) \geq h_i\}$ // initialize set with seed nodes
while $Q \neq \emptyset$ **do**
    select $i \in Q$ and remove from Q
    $V^{\mathcal{S}} = V^{\mathcal{S}} \cup \{i\}$
    foreach $(i,j) \in A : j \notin V^{\mathcal{S}}$ **do**
        $N_j^{\mathcal{S}} = N_j^{\mathcal{S}} \cup \{i\}$
        if $f_j(N_j^{\mathcal{S}}, p_j) \geq h_j$ **then** $Q = Q \cup \{j\}$
return $\mathcal{S} = (V^{\mathcal{S}}, N^{\mathcal{S}}, p)$

and removed from Q. Cycles cannot occur since no influence is exerted to already activated nodes. Thus, the algorithm requires $O(|A|)$ evaluations of activation functions f_i, i.e., it runs in polynomial time for fixed incentives p. □

2 Literature Review

Domingos and Richardson [6, 20] were among the first who considered network effects in an influence maximization problem, where a small set of influential users (i.e., a *target set*) shall be targeted initially to trigger a cascade of adoptions. A probabilistic model (Markov random field) is used for the interactions between customers and heuristic solution methods are proposed. Building upon the results by Domingos and Richardson [6], Kempe et al. [14] introduce a discrete optimization problem that they call the influence maximization problem. The goal is to identify an initially active node set of size k that maximizes the (expected) number of finally activated nodes. This problem and some closely related variants are also called *Target Set Selection Problem (TSSP)*. Kempe et al. [14] consider two basic diffusion models: (i) a *linear threshold model* based on uniformly distributed thresholds, and (ii) a probabilistic *independent cascade model*. They show that the problem is NP-hard and that a simple greedy heuristic adding the element with highest marginal gain in each step yields an approximation ratio of $1 - 1/e - \varepsilon$ under certain conditions. They also show that it is NP-hard to approximate the problem within a factor of $|V|^{1-\varepsilon}$ when using general threshold functions. Further approximation results for special cases and the decreasing cascade model which generalizes the independent cascade model, are introduced in [15, 16].

Chen [4] considers a variant of the TSSP which seeks for a target set of minimum size such that at least a given fraction of nodes will be activated. In contrast to Kempe et al. [14], deterministic thresholds are given for each node and the influence of each neighbor is equal to one. Chen [4] shows that it is NP-hard to approximate this problem within a polylogarithmic factor and proposes an exact polynomial time algorithm for the case when the input graph is an undirected tree. Ben-Zwi et al. [2] generalize the latter result by proposing an exact algorithm for graphs with bounded treewidth ω that has a runtime of $|V|^{\mathcal{O}(\omega)}$. Ackerman et al. [1] were among the first who propose an ILP formulation for the TSSP. Their formulation uses arc and node design variables and completes the input graph by non-edges (a technique that is typically applied in ordering problems). A cycle-free propagation graph is then ensured through triangle elimination constraints. We observe that the variant of the TSSP studied in [1, 2, 4] is a special case of the GLCIP with additively separable

activation functions, $P_i = \{0, h_i\}$, and $w_{ih_i} = 1$ for all $i \in V$. Raghavan and Zhang [19] introduce the *Weighted Target Set Selection Problem* (WTSSP) on undirected graphs which generalizes the TSSP by introducing (node-dependent) costs for initially activating nodes. To obtain the WTSSP as a special case of the GLCIP, we set the incentive costs for activating a node to the node-dependent costs and introduce two oppositely directed arcs (with influence equal to one) for each undirected edge. A polynomial time algorithm as well as a tight and compact ILP formulation for the WTSSP on trees is given. The main idea here is to split each edge by introducing a dummy node and consider four variables for each original edge that indicate exerted influence (in addition to classical node and arc design variables). Raghavan and Zhang [19] show that this extended formulation is tight if the input graph is a tree, while the LP relaxation of an analogous one based on usual node and arc design variables has fractional extreme points. Subsequently, this new formulation is extended to the case of general graphs (in which case the LP relaxation may produce a fractional solution) by adding cycle elimination constraints. Note that the strength of their formulation is (also) based on the fact that at least one ingoing arc needs to be selected for each dummy node. Thus, one cannot extend this idea (in a straightforward manner) for directed graphs while still ensuring that the solution does not contain cycles.

Another closely related problem is the *Least Cost Influence Problem* (LCIP) which generalizes the TSSP by introducing partial incentives that reduce a node's hurdle and enabling node activations by combining incentives with influence from neighbors. In contrast to the (W)TSSP, the strength of influence may differ among neighbors of a node in the LCIP. The LCIP which aims to minimize the sum of offered incentives while activating at least a predefined number of nodes is introduced by Günneç [12] and further investigated in [11, 13]. It is a special case of the GLCIP in which every incentive $p \in [0, h_i]$ with costs $w_{ip} = p$ can be paid to node i and $f_i(U, p) = p + \sum_{j \in U} d_{ji}$ holds for all $U \in \mathcal{N}_i$ and $p \in P_i$. Günneç et al. [13] propose a time-indexed formulation, show that the problem can be solved in polynomial time if the input graph is a tree and all neighbors of a node exert identical influence on it. Moreover, they prove that the LCIP is NP-hard in general and in further special cases (including the case of tree graphs with unequal influence). Subsequently, they focus on the case of equal influence from each node's neighbor. A main observation is that the number of influencing neighbors g_i needed to activate node i without additional incentives can be precomputed. Based on this, a sophisticated ILP formulation with three variables per arc indicating full, partial and no influence is derived. For each node at most $g_i - 1$ neighbors may exert their full influence, while at most one may exert partial influence. An ILP formulation based on this idea that uses cycle elimination constraints is proposed which has a tight linear programming (LP) relaxation if the input graph is a tree. The requirement that the needed number of influencing nodes does not depend on the set of chosen nodes makes it unlikely, however, that their formulation can be (easily) extended to the case of unequal influence.

Wu and Küçükyavuz [22] study two-stage stochastic optimization problems in which the second-stage objective function is submodular, and propose an exact method based on delayed constraint generation. A computational study is performed on stochastic variants of the influence maximization problem that are based on the independent cascade and linear threshold models. Moreover, several new problems aiming to identify key players in social networks, variants of influence maximization problems, and heuristic algorithms for solving them have been proposed recently, see, e.g., Borgatti [3], Kimura et al. [17].

We conclude that none of the more sophisticated formulations proposed for the WTSSP [19] and the LCIP [13] can be re-used with slight adaptations for the GLCIP with additively separable

and linear activation functions, and that the case of general activation functions has not been considered yet in the operations research community. On the contrary, the above mentioned time-indexed formulation for the LCIP, as well as the formulation by Ackerman et al. [1] for the TSSP, can be extended in a straightforward manner to the GLCIP with additively separable and linear activation functions. The GLCIP is NP-hard as it contains the NP-hard LCIP [13] as a special case.

3 Set covering formulation

In this section, we propose a set covering formulation for the GLCIP with arbitrary activation functions. To the best of our knowledge, this is the first ILP formulation of that kind, since all previous formulations are restricted to special cases of the GLCIP and to additively separable and linear threshold functions. Our set covering ILP formulation is based on introducing one variable for each *minimal influencing set* for each node, a concept that is formally defined in Definition 5. We further show in Theorem 2 that it suffices to restrict the attention to these minimal influencing sets, instead of considering all subsets of neighbors of some node.

Definition 5 (Minimal Influencing Set)**.** *Let $U \subseteq N_i$ be a set of active neighbors of node $i \in V$, such that there exists an incentive $p \in P_i$ which suffices to activate it, i.e., $f_i(U,p) \geq h_i$. Furthermore, let $p' = \min\{\tilde{p} \in P_i \mid f_i(U,\tilde{p}) \geq h_i\}$ be the minimum incentive (possibly equal to zero) required to achieve that activation. Then, U is a* minimal influencing set *for node $i \in V$ if and only if there does not exist a set $U' \subset U$ such that $f_i(U',p') \geq h_i$, i.e., node i cannot be activated by a proper subset of these neighbors with the same incentive. For each node $i \in V$, let $\Lambda_i \subseteq \mathcal{N}_i$ denote the set of all minimal influencing subsets.*

Theorem 2. *There exists an optimal solution $\mathcal{S} = (V^{\mathcal{S}}, N^{\mathcal{S}}, p^{\mathcal{S}})$ such that every active node $i \in V^{\mathcal{S}}$ is influenced by a minimal influencing set $N_i^{\mathcal{S}} \in \Lambda_i$.*

Proof. Consider a node $u \in V^{\mathcal{S}}$ which is not influenced by a minimal influencing set in solution $\mathcal{S}$, i.e., $N_u^{\mathcal{S}} \notin \Lambda_u$. Then, there exists a subset $U \subset N_u^{\mathcal{S}}$ such that $f_u(U, p_u^{\mathcal{S}}) \geq h_u$ and $U \in \Lambda_u$. Clearly, $\mathcal{T} = (V^{\mathcal{T}}, N^{\mathcal{T}}, p^{\mathcal{T}})$ such that $V^{\mathcal{T}} = V^{\mathcal{S}}$, $p^{\mathcal{T}} = p^{\mathcal{S}}$, $N_i^{\mathcal{T}} = N_i^{\mathcal{S}}$, $\forall i \in V^{\mathcal{T}} \setminus \{u\}$, and $N_u^{\mathcal{T}} = U$, is a solution with the same objective value as $\mathcal{S}$ in which u is influenced by a minimal influencing set. The results follows by repeating this argument for every node that is not activated by a minimal influencing set. □

Formulation (1), to which we will refer to as (COV), uses the following three sets of variables: (i) node variables $x_i \in \{0,1\}$ for all $i \in V$ that indicate whether node i is activated; (ii) arc variables $z_{ij} \in \{0,1\}$ for all $(i,j) \in A$ that indicate arcs on which influence is exerted; and (iii) influencing subset variables $\lambda_i^U \in \{0,1\}$ for all $i \in V$ and $U \in \Lambda_i$ that indicate whether a minimal influencing set U is used (together with an appropriate incentive) in the activation of node i. In the following, we use $p_i^U = \operatorname{argmin}_{p \in P_i}\{f_i(U,p) \geq h_i\}$ to denote the (unique) minimum incentive to activate node i when influenced by set U and use costs $w_i^U = w_{ip_i^U}$ to denote the associated costs; cf. Proposition 2 for the efficient computation of p_i^U.

$$\text{(COV)} \quad \min \quad \sum_{i \in V} \sum_{U \in \Lambda_i} w_i^U \lambda_i^U \tag{1a}$$

$$\text{s.t.} \quad \sum_{U \in \Lambda_i} \lambda_i^U = x_i \qquad \forall i \in V \tag{1b}$$

$$\sum_{U \in \Lambda_j : i \in U} \lambda_j^U = z_{ij} \qquad \forall (i,j) \in A \tag{1c}$$

$$\sum_{(i,j) \in C} z_{ij} \leq \sum_{i \in V(C) \setminus \{k\}} x_i \qquad \forall k \in V(C), \forall \text{ cycles } C \subseteq A \tag{1d}$$

$$z_{ij} \leq x_i \qquad \forall (i,j) \in A \text{ s.t. } (j,i) \notin A \tag{1e}$$

$$\sum_{i \in V} x_i \geq \lceil \alpha |V| \rceil \tag{1f}$$

$$x_i \in \{0,1\} \qquad \forall i \in V \tag{1g}$$

$$z_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{1h}$$

$$\lambda_i^U \geq 0 \qquad \forall i \in V,\ \forall U \in \Lambda_i. \tag{1i}$$

The objective function (1a) minimizes the cost occurring for offered incentives. Recall that for each $U \in \Lambda_i$, the costs w_i^U are constant and that $w_i^U = 0$ if set U is sufficient to activate node i without additional incentive. Propagation constraints (1b) state that exactly one influencing set needs to be selected for each activated node. Equations (1c) ensure that all arcs on which influence is exerted are chosen as well. *Generalized cycle-elimination constraints* (1d) where $V(C) = \{i \in V \mid (i,j) \in C\}$ ensure that the subgraph induced by the set of arcs on which influence is exerted is acyclic; cf. Definition 4. Linking constraints (1e) ensure that influence can only be exerted on arc $(i,j) \in A$ if node i is active. Notice that the latter condition is enforced by constraints (1d) for all arcs $(i,j) \in A$ such that $(j,i) \in A$. Coverage constraint (1f) accounts for the minimum fraction of nodes that need to be activated. Observe that only lower bounds are imposed for influencing subset variables in (1i). They will, however, become binary automatically due to the other constraints. Further note that we refrain from eliminating node and arc variables using equations (1b) and (1c) to avoid the need for developing problem specific branching rules.

3.1 Generalized propagation constraints

In this section, we propose strengthening inequalities for (COV) and show that they dominate generalized cycle elimination constraints (1d). Their validity is based on the observation that for every set of nodes $X \subseteq V$ containing an active node i, at least one of the following two conditions must hold:

(i) at least one node $j \in X$ is activated via a minimal influencing set $U \in \Lambda_j$ (together with incentive p_j^U) that does not contain nodes from X, i.e., $U \cap X = \emptyset$,

(ii) at least one node $j \in X$ receives incentive $p_j^\emptyset$, i.e., is activated without receiving influence from neighboring nodes.

If none of the two conditions holds, the subgraph induced by the set of chosen arc variables induces a cycle inside X due to linking constraints (1c). These observations are captured by the following *generalized propagation constraints*:

$$\sum_{j \in X} \sum_{U \in \Lambda_j : U \cap X = \emptyset} \lambda_j^U \geq x_k \qquad \forall k \in X, \forall X \subseteq V. \tag{2}$$

Note that variables $\lambda_j^{\emptyset}$ corresponding to activation without external influence are included in the sum on the left-hand side, and observe that generalized propagation constraints correspond to propagation constraints (1b) rewritten in greater than or equal form if X is a singleton, i.e., $X = \{k\}$. One intuitive reason why they can strengthen the LP relaxation of formulation (COV) if $|X| \geq 2$ is the absence of variables whose associated minimal influencing sets intersect with X on the left-hand side. Further observe that the right-hand side of inequalities (2) can be lifted to one if at least one node from X needs to be activated, i.e., in case $|X| > \lfloor (1-\alpha)|V| \rfloor$. In this case only one inequality per set $X \subseteq V$ is considered. Proposition 3 reveals dominance between some particular generalized propagation constraints.

Proposition 3. *Consider node $k \in V$ and node sets X and Y such that $X \subset Y \subseteq V$ and $k \in X$. Assume that the subgraph induced by Y does not contain arcs from nodes in Y to nodes in X. Then, the generalized propagation constraint associated with node set X dominates the one associated with node set Y.*

Proof. The result follows from the following chain of inequalities:

$$\sum_{j \in Y} \sum_{U \in \Lambda_j : U \cap Y = \emptyset} \lambda_j^U \geq \sum_{j \in X} \sum_{U \in \Lambda_j : U \cap Y = \emptyset} \lambda_j^U = \sum_{j \in X} \sum_{U \in \Lambda_j : U \cap X = \emptyset} \lambda_j^U \geq x_k.$$

Thereby, the equation holds since $(Y \setminus X) \cap (\cup_{i \in X} N_i) = \emptyset$ (by assumption) and hence $U \cap Y = \emptyset$ iff $U \cap X = \emptyset$ for $U \in \Lambda_j$ such that $j \in X$. □

An immediate consequence of Proposition 3 is that we only need to consider generalized propagation constraints such that there exists a path in X from every node in X to node k chosen on the right-hand side of the inequality.

Theorem 3. *Generalized propagation constraints (2) dominate generalized cycle elimination constraints (1d).*

Proof. Consider a cycle $C \subseteq A$ with node set $V(C)$ and a fixed node $k \in V(C)$. Then, the associated generalized propagation constraint together with equations (1b) and (1c) implies that:

$$\begin{aligned} x_k &\leq \sum_{i \in V(C)} \sum_{U \in \Lambda_i : U \cap V(C) = \emptyset} \lambda_i^U \leq \sum_{i \in V(C)} \sum_{U \in \Lambda_i : \nexists j \in U : (j,i) \in C} \lambda_i^U = \\ &= \sum_{i \in V(C)} \sum_{U \in \Lambda_i} \lambda_i^U - \sum_{i \in V(C)} \sum_{U \in \Lambda_i : \exists j \in U : (j,i) \in C} \lambda_i^U = \sum_{i \in V(C)} x_i - \sum_{(i,j) \in C} z_{ij}. \end{aligned}$$

Thereby, the second inequality holds since the set of influencing subset variables on the right hand side is a superset of the one on the left hand side. The theorem follows from rearranging the terms of the outer inequality. □

This result suggests to heuristically separate constraints (2) by searching for violated cycle elimination constraints. This and further separation heuristics will be detailed in Section 5.

3.2 Pricing subproblem

The number of minimal influencing sets may grow exponentially with the indegree of each node. We will therefore use column generation to dynamically generate minimal influencing set variables for instances in which the large number of such variables prohibits simply including all of them explicitly in the (initial) model. Observe that (COV) remains valid when rewriting equations (1b) in $\geq$ form, and rewriting (1c) in $\leq$ form. Then, by replacing all $\leq$ constraints by $\geq$ inequalities and by arranging all variables on their left hand side, we obtain dual variables $\mu_i \geq 0, \forall i \in V$, $\pi_{ij} \geq 0, \forall (i,j) \in A$, and $\phi_i^X \geq 0, \forall X \subseteq V, i \in X$, associated to constraints (1b), (1c), and (2), respectively. For node $i \in V$ and influencing set U, we will also use $\Phi_i^U = \{(X,k) \mid X \subseteq V \setminus U, i,k \in X\}$ to denote all pairs (X,k) for which the dual variable of the generalized propagation constraint associated to node set X and node k needs to be considered to calculate the reduced costs of variables λ_i^U. Then, the pricing subproblem associated to node $i \in V$, can be formally stated as

$$U^* = \text{argmin}_{U \in \Lambda_i} \{w_i^U - \mu_i + \sum_{j \in U} \pi_{ji} - \sum_{(X,k) \in \Phi_i^U} \phi_k^X\}. \tag{3}$$

As will be detailed in Section 5.3, the pricing subproblem for node $i \in V$ is equivalent to the NP-hard knapsack problem [8] (in its minimization form) for additively separable and linear activation functions in case (a) we do not consider generalized propagation constraints, and (b) we further assume that at most one non-zero incentive exists. Thus, Theorem 4 follows.

Theorem 4. *The pricing subproblem (3) associated to node $i \in V$ is NP-hard.*

4 Arc formulation

Alternative formulations that model the propagation with arc variables have been proposed for several related problems, see, e.g., Ackerman et al. [1], Günneç et al. [13]. Their validity is based upon the fact that they ensure a directed acyclic propagation graph, cf. Definition 4. Formulation (4), to which we will refer to as (ARC), generalizes similar arc formulations from the literature to the special case of the GLCIP in which all activation functions are additively separable and linear. We use this formulation for the computational comparison in Section 6 and for the generation of the initial set of variables for model (COV), cf. Section 5.5. As above, node variables $x_i \in \{0,1\}$, $\forall i \in V$, indicate whether or not a node i is activated and arc variables $z_{ij} \in \{0,1\}$, $\forall (i,j) \in A$, indicate on which arcs influence is exerted. In addition, variables $x_{ip} \in \{0,1\}$, $\forall i \in V$, $\forall p \in P_i$, indicate the incentive received by a node.

$$\text{(ARC)} \quad \min \quad \sum_{i \in V} \sum_{p \in P_i} w_{ip} x_{ip} \tag{4a}$$

$$\text{s.t.} \quad \sum_{p \in P_i} p x_{ip} + \sum_{(j,i) \in A} d_{ji} z_{ji} \geq h_i x_i \qquad \forall i \in V \tag{4b}$$

$$\sum_{p \in P_i} x_{ip} = x_i \qquad \forall i \in V \tag{4c}$$

$$\sum_{(i,j) \in C} z_{ij} \leq \sum_{i \in V(C) \setminus \{k\}} x_i \qquad \forall k \in V(C), \forall \text{ cycles } C \subseteq A \tag{4d}$$

$$z_{ij} \leq x_i \qquad \forall (i,j) \in A \text{ s.t. } (j,i) \notin A \tag{4e}$$

$$\sum_{i \in V} x_i \geq \lceil \alpha |V| \rceil \tag{4f}$$

$$x_{ip} \in \{0,1\} \qquad \forall i \in V, \forall p \in P_i \tag{4g}$$

$$x_i \in \{0,1\} \qquad \forall i \in V \tag{4h}$$

$$z_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{4i}$$

Objective function (4a) minimizes the sum of costs for selected incentives. Note that non-linear incentive cost functions can be used since one variable exists for each possible incentive. Propagation constraints (4b) compare the sum of chosen incentives and the influence coming from neighbors on ingoing arcs to a node's hurdle. Constraints (4c) ensure that exactly one incentive is given to each activated node. Generalized cycle elimination constraints (4d), forcing constraints (4e), and the coverage constraint (4f) are identical to previously discussed inequalities, cf. Section 3.

5 Algorithmic framework

This section first details our algorithms for the dynamic separation of violated generalized cycle elimination constraints (Section 5.1) and generalized propagation constraints (Section 5.2). Section 5.3 describes how we solve the pricing subproblem for minimal influencing set variables, while Section 5.4 introduces the initial and primal heuristics used in our implementation. Finally, Section 5.5 details our heuristic solution approach based on column generation. Throughout this section, let $\bar{x}_j$, $\forall j \in V$, $\bar{z}_{ij}$, $\forall (i,j) \in A$, and $\bar{\lambda}_i^U$, $\forall i \in V$, $\forall U \in \Lambda_i$, denote the current variable values of node, arc, and minimal influencing set variables, respectively.

5.1 Separation of generalized cycle elimination constraints

We adapt the shortest path algorithm in Grötschel et al. [10] for the separation of classical cycle elimination cuts by using weights $w_{ij} := \bar{x}_i - \bar{z}_{ij} \geq 0$ for all $(i,j) \in A$. From each node $k \in V$ we compute shortest paths to all neighbors N_k. If the total weight of a shortest path from k to $j \in N_k$ together with arc (j,k) (forming a cycle) is less than $\bar{x}_k$, we found a violated generalized cycle elimination constraint (1d). Except for comparing LP relaxation bounds, we skip the separation in case $(\bar{x}, \bar{z})$ is fractional for two reasons: (i) while a large number of inequalities (1d) is often violated, adding them to the model rarely improves the LP bound; (ii) the dominance result in Theorem 3 suggests to focus on generalized propagation constraints, see Section 5.2. To cut off infeasible integer solutions, we search for cycles in the support graph defined by $\bar{x}$ and $\bar{z}$ with breadth-first search.

5.2 Separation of generalized propagation constraints

We first state an ILP whose solution corresponds to a maximally violated generalized propagation constraint (2) in case the objective value is negative. Formulation (5) uses variables $v_i \in \{0,1\}$, $\forall i \in V$, that indicate membership of nodes to set X and $u_i^U \in \{0,1\}$, $\forall i \in V$, $\forall U \in \Lambda_i$, that are equal to one iff node $i \in X$ and $U \cap X = \emptyset$, in which case variable λ_i^U is part of the left-hand side of inequality (2). Variable $y_i \in \{0,1\}$, $\forall i \in V$, indicates whether node i is on the right-hand side of a generalized propagation constraint. Finally, variable $r \in \{0,1\}$ decides whether the lifting to one on the right-hand side is applied or not. Observe that it is sufficient to consider variables v_i

for nodes $i \in V$ such that $\bar{x}_i > 0$ and variables u_i^U for sets $U \in \Lambda_i$, $i \in V$, such that $\bar{\lambda}_i^U > 0$. We assume that the remaining variables are fixed to zero in order to simplify notation.

$$\min \quad \sum_{i \in V} \left(\sum_{U \in \Lambda_i} \bar{\lambda}_j^U u_i^U - \bar{x}_i y_i \right) - r \tag{5a}$$

$$\sum_{i \in V} v_i \geq 2 + (\lfloor (1-\alpha)|V| \rfloor - 2) r \tag{5b}$$

$$\sum_{i \in V} y_i + r = 1 \tag{5c}$$

$$y_i \leq v_i \qquad \forall i \in V \tag{5d}$$

$$v_i \leq \sum_{j \in U} v_j + u_i^U \qquad \forall i \in V,\ \forall U \in \Lambda_i \tag{5e}$$

$$v_i, y_i \in \{0,1\} \qquad \forall i \in V \tag{5f}$$

$$u_i^U \in \{0,1\} \qquad \forall i \in V,\ \forall U \in \Lambda_i \tag{5g}$$

$$r \in \{0,1\}. \tag{5h}$$

The objective function (5a) maximizes the constraint violation, and each solution with a negative value corresponds to a violated generalized propagation constraint (2). Inequalities (5b) force a minimum size of two for set X (since equalities (1b) cover the case of $|X| = 1$) or size $\lfloor (1-\alpha)|V| \rfloor$ if the lifting is applied. Equalities (5c) together with (5d) decide for the node or the lifting on the right-hand side of the inequality. Finally, forcing constraints (5e) ensure that all variables u_i^U corresponding to relevant minimal influencing sets (i.e., $i \in X$ and $U \cap X = \emptyset$) are set to one.

To separate inequalities (2) heuristically, we propose three different methods detailed in the following: (i) a heuristic based on finding cycles, (ii) a greedy set extension heuristic, and (iii) a greedy set reduction heuristic.

The dominance result in Theorem 3 suggests a heuristic based on the exact separation of generalized cycle elimination constraints, see Section 5.1: For each found cycle C we check the violation of the generalized propagation constraint (2) with $X = V(C)$. For the right-hand side we choose node $k := \operatorname{argmax}_{j \in X}\{\bar{x}_j\}$. The greedy set extension heuristic is based on Proposition 3 and starts from $X = \{k\}$ for every node $k \in V$ with $\bar{x}_k > 0$ and $\bar{\lambda}_k^{\emptyset} < \bar{x}_k$: We iteratively build a candidate list $L := \{j \in V \setminus X : \delta^+(j) \cap \delta^-(X) \neq \emptyset\}$ and extend X by adding the node $j \in L$ which leads to the smallest left-hand side of inequality (2). At each iteration we check whether inequality (2) for set X and node k is violated, in which case we add it to the model. Finally, the idea of the greedy set reduction heuristic is to consider very large sets X because of two reasons: (i) inequalities (2) tend to be sparse for large sets X since there are in general not many set variables corresponding to influence from outside X, and (ii) sets X with cardinality $|X| \geq \lfloor (1-\alpha)|V| \rfloor$ allow the lifting to 1 on the right-hand side. We start with set $X = V$ and iteratively remove the node i with largest value $\sum_{U \in \Lambda_i : U \cap X = \emptyset} \bar{\lambda}_i^U$ as long as $|X| \geq \lfloor (1-\alpha)|V| \rfloor$. In each iteration we check for each node $j \in X$ if its removal would lead to a violated propagation constraint, in which case we add it to the model.

Similar to generalized cycle elimination constraints, in the integral case we search for a cycle C in the support graph defined by $\bar{x}$ and $\bar{z}$ with breadth-first search and add inequality (2) for $X = V(C)$ and an arbitrary node $k \in V(C)$.

Algorithm 2: Dynamic program for solving the pricing subproblem.

// Assumptions: neighbors N_i are ordered (arbitrarily) $\{k_1, k_2, \ldots, k_{|N_i|}\}$
// $c_j(d, \Phi) = \infty$ if $(d, \Phi) \notin L_j$
$L_0 = \{(0, \Phi_i^\emptyset)\}$ // initialization (no influence from neighbors)
$c_0(0, \Phi_i^\emptyset) = \hat{w}_i(0) - \mu_i - \sum_{(X,k)\in\Phi_i^\emptyset} \phi_k^X$ // initial reduced costs
for $j \in \{1, \ldots, |N_i|\}$ **do**
  forall $(d, \Phi) \in L_{j-1}$ **do**
    $c_j(d, \Phi) = \min\{c_j(d, \Phi), c_{j-1}(d, \Phi)\}$ // no influence from k_j
    $L_j = L_j \cup \{(d, \Phi)\}$
    $\Phi' = \{(X, k) \in \Phi : j \notin X\}$ // relevant sets after adding k_j
    $c_j(d + d_{k_j i}, \Phi') = \min\{c_j(d + d_{k_j i}, \Phi'), c_{j-1}(d, \Phi) +$
      $+ \sum_{(X,k)\in\Phi\setminus\Phi'} \phi_k^X + \pi_{ji} + \hat{w}_i(d + d_{ji}) - \hat{w}(d)\}$
    $L_j = L_j \cup \{(d + d_{k_j i}, \Phi')\}$
    if $c_j(d + d_{k_j i}, \Phi') < 0$ **then** subset with negative reduced cost found

5.3 Solving the pricing subproblem

For the additively separable and linear case, the relation between the pricing subproblem without generalized propagation constraints (2) and the knapsack problem stated in Section 3.2 implies the existence of a pseudo-polynomial solution algorithm for the pricing subproblem if all influence weights are integral. Algorithm 2 describes a corresponding dynamic program that considers the set $\Phi_i^\emptyset$ of separated generalized propagation constraints (2) and more general activation functions of the form $f_i(U, p) = \left(\sum_{j\in U} d_{ji}\right)^\Gamma + p$, $i \in V$, $U \in \Lambda_i$, $p \in P_i$, that will be used in our computational experiments. It considers set $\{k_1, k_2, \ldots, k_{|N_i|}\}$ of all neighbors from i, ordered arbitrarily (here by increasing $\pi_{k_j i}/d_{k_j i}$), and computes the minimal reduced costs $c_j(d, \Phi)$ among all subsets $U \subseteq \{k_1, k_2, \ldots, k_j\}$ of the first j neighbors, $1 \le j \le |N_i|$, for all sets $\Phi \subseteq \Phi_i^\emptyset$ corresponding to generalized propagation constraints whose dual variable values need to be considered if i is influenced by U, i.e., $\Phi = \{(X, k) \in \Phi_i^\emptyset \mid X \cap U = \emptyset\}$. Algorithm 2 also uses $\hat{w}_i(d) = \min_{p\in P_i}\{w_{ip} \mid d^\Gamma + p \ge h_i\}$ to denote the cheapest incentive to activate node i when the sum of neighboring influences is equal to d. Its validity follows from the fact that a set $U \subseteq \{k_1, k_2, \ldots, k_j\}$, $1 \le j \le |N_i|$, cannot yield an optimal solution (after considering the remaining neighbors $k_{j+1}, \ldots, k_{|N_i|}$) if there exists a set $U' \subseteq \{k_1, k_2, \ldots, k_j\}$ such that $\sum_{j\in U} d_{ji} = \sum_{j\in U'} d_{ji}$, $\Phi_i^U = \Phi_i^{U'}$, and $w_i^U + \sum_{j\in U} \pi_{ji} - \sum_{(X,k)\in\Phi_i^U} \phi_k^U > w_i^{U'} + \sum_{j\in U'} \pi_{ji} - \sum_{(X,k)\in\Phi_i^{U'}} \phi_k^{U'}$.

Finally, observe that Algorithm 2 boils down to the classic and well known dynamic program (presented as forward recursion) for the knapsack problem if no generalized propagation constraints need to be considered, i.e., if $\Phi_i^\emptyset = \emptyset$. In this case only the most efficient partial solution is stored and extended for each relevant subset of neighbors and sum of their influence, respectively.

5.4 Initial and Primal Heuristics

We use three simple greedy construction heuristics to obtain feasible solutions (i) to get an initial solution and primal bound, (ii) to warm-start column generation with an initial set of set variables, and (iii) to improve primal bounds throughout the solution process by exploiting LP information. The heuristics *MinIncentive* and *MinInfluence* have been proposed by Günneç et al. [13] and are (slightly) adapted to also deal with unequal influences, discrete sets of incentives, general activation functions, and arbitrary lower bounds on the number of nodes to activate. Common to all greedy construction heuristics described here is the following procedure: We start with an empty solution; at each iteration we activate a not yet active node by paying the minimum available incentive to reach its hurdle, taking into account the current influence coming from already active neighbors. After activating the chosen node, we continue propagation by exerting its influence to all inactive neighbors and iteratively continue propagation from all activated nodes, i.e., a version of Algorithm 1 is applied in which the propagation graph from the previous iteration is considered as starting solution. The heuristics only differ in the greedy criterion to select the next node to activate:

- *MinIncentive*: The node with minimal incentive to activate it is chosen.
- *MinInfluence*: The node $i \in V$ with minimal average influence over all neighbors N_i is chosen; the *MinIncentive* greedy value for a node is used instead if it is lower than the average influence in the current iteration.
- *MaxActivation*: The *MinIncentive* greedy value divided by the number of neighbors which would be activated in the next propagation step, is used.

An LP solution is taken into account by reducing the *MinIncentive* greedy value by the incentive given to the node in the LP solution. Ties are broken by selecting the node with maximal number of non-active neighbors on outgoing arcs. All heuristics are applied sequentially and the best solution found is kept. Additionally, all minimal influencing set variables are added whose corresponding sets are contained in such a solution and which have not been included in the model yet.

5.5 Price-Cut-and-Branch

Next, we propose a heuristic algorithm based on column generation that allows to tackle instances for which one cannot simply enumerate all minimal influencing set variables and explicitly include them in the initial model. Our heuristic first applies a *column and cut generation* procedure intended to obtain tight primal and dual bounds before switching to a *branch-and-cut* phase whose sole purpose is to further improve the primal bound. In the first phase the LP relaxation of model (COV) is solved by column generation together with separation of generalized propagation constraints (2). The initial set of variables is defined by variables corresponding to (i) empty influencing sets, (ii) minimal influencing sets used in a solution found by any of the three greedy heuristics detailed in Section 5.4, and (iii) mimimal influencing sets generated from the LP solution of model (ARC) which can usually be solved in comparably short computation time. The latter variables are generated as follows. For each node $j \in V$ we sort all arcs $(i, j) \in A$ with $\bar{z}_{ij} > 0$ by descending LP value. We iteratively take arcs in the given order and build a set of influencing neighbors until either there is no arc left in the list or we obtain an influencing set without need for additional incentives. After reducing it to a minimal influencing set U (if necessary), we add it to

the model. Then, we reduce the LP values of all arcs which are involved in set U by $\min_{i \in U} \bar{z}_{ij}$ and repeat this process until the sum of the latter reductions are at least equal to one. At each pricing iteration we run all greedy heuristics guided by the current LP solution of the restricted master problem to potentially obtain new incumbent solutions, and generate new columns corresponding to minimal influencing sets in the solutions. If no further columns with negative reduced costs exist, we perform a fixed number of cut separation rounds and then return to pricing. This sequence is repeated until no further relevant columns and violated inequalities are found. In this way, we obtain both a lower and an upper bound to the optimal solution value. Based on the set of columns and cuts found in the first phase we start a classical branch-and-cut algorithm without adding further columns and generalized propagation constraints (2) to the model. The only purpose of this phase is to improve the primal bound.

6 Computational Study

In this section we describe the computational environment, report detailed experimental results, and finally discuss the outcome. Each experiment has been performed on a single core of an Intel Xeon E5-2670v2 machine with 2.5 GHz. A time limit of 7 200 seconds and a memory limit of 8 GB has been set. The algorithms are implemented in C++ and IBM ILOG CPLEX 12.7.1 is used as branch-and-cut framework and LP solver. Unless otherwise stated, CPLEX parameters are set to their defaults.

Benchmark Instances and Activation Functions. Instance set SW has been generated in the following way: First, we create five directed small-world graphs [21] for each node-set size $|V| \in \{50, 75, 100\}$, average node degree $k \in \{4, 8, 12, 16\}$, and rewiring probability $\beta \in \{0.1, 0.3\}$. Influence values d_{ij} on arcs $(i, j) \in A$ are distributed uniformly random in $\{1, \ldots, 10\}$. Let $D_i = \sum_{j \in N_i} d_{ji}$ be the total amount of influence possibly exerted on node i from its neighbors. Then, $h_i = \max\{1, \min\{\zeta_i, D_i\}\}$ rounded to the next integer, where $\zeta_i \sim N(0.7D_i, D_i/|N_i|)$ is a normally-distributed random variable. The minimal fraction of nodes to activate is set to $\alpha \in \{0.1, 0.5, 1\}$. For each node $i \in V$ we use activation function $f_i(U, p) = \left(\sum_{j \in U} d_{ji}\right)^{\Gamma} + p$. For $\Gamma = 1$, f_i is additively separable and linear. Values $\Gamma > 1$ model, e.g., the situation of peer pressure where additional influence has an over-proportional effect. On the contrary, values $\Gamma < 1$ lead to a submodular function and model the situation of diminishing marginal influences, cf. Leskovec et al. [18]. Here, we consider $\Gamma \in \{0.9, 1, 1.1\}$. The discrete set of incentives is based on the maximal hurdle $\hat{h} = \max_{i \in V} h_i$ and is defined as $P_i = \{0, 0.25\hat{h}, 0.5\hat{h}, 0.75\hat{h}, \hat{h}\}$ for all $i \in V$. The costs for one particular incentive is not necessarily proportional to the incentive value. Thus, assuming economies of scale, we define $w_{ip} := p^{0.9}$. Instance set GRZ has been created in the same way as the instances from Günneç et al. [13]. Thus, these instances enable a comparison of our solution methods to their approach for the special case of the GLCIP with additively separable and linear activation functions, $\alpha = 1$, equal influences for all incoming arcs of a node, and continuous incentives (i.e., $P_i = [0, h_i]$) for which $w_{ip} = p$, $\forall i \in V$, $\forall p \in P_i$, holds.

6.1 Algorithms and Settings.

The following algorithms (and variants thereof) are considered in our study:

G: A re-implementation of the state-of-the-art approach from Günneç et al. [13] for the special case of $\alpha = 1$, equal influence values for all neighbors N_i of node $i \in V$, additively separable and linear activation functions, and continuous incentives $P_i = [0, h_i]$ with $w_{ip} = p$, $\forall i \in V$, $\forall p \in P_i$. Preliminary experiments showed that a parametrization which differs from Günneç et al. [13] is beneficial, see below. We use our primal heuristics and we statically include in the initial model the generalized cycle elimination constraints (1d) for all cycles up to length four, while separation of the remaining inequalities is only done in the integral case, cf. Section 5.1.

A: This branch-and-cut algorithm is based on model (ARC), see Section 4, and can only be applied for additively separable and linear activation functions. Generalized cycle elimination constraints (1d) for all cycles up to length four are added a priori to the model, while separation of the remaining inequalities is only done in the integral case, cf. Section 5.1.

C: An exact method based on model (COV) which first enumerates all set variables, adds them to the initial model, and then performs a branch-and-cut. Inequalities (2) are separated only in the integer case.

P: The price-cut-and-branch heuristic described in Section 5.5.

Further variants of algorithms C and P are obtained by including (i) heuristic separation (variants C^+ and P^+) or (ii) heuristic and exact separation (variants C_{e}^+ and P_{e}^+) of generalized propagation constraints (2). All algorithms use the heuristics described in Section 5.4.

Separation of generalized propagation constraints We set a limit of 100 violated inequalities per iteration, avoiding duplicates by a hash table. To accelerate the separation methods, in the beginning we restrict set X to a maximum of five nodes and in case of the greedy set reduction heuristic to a minimum of $|V| - 5$ nodes. Inequalities (2) based on small and large sets tend to be sparse and seem to be beneficial for cut convergence. If less than 50 violated inequalities are found in some iteration, the value limiting the cardinality of set X is doubled. Algorithms C_{e}^+ and P_{e}^+ apply the exact ILP-based separation if less than 50 cuts are found by the separation heuristics in the same iteration. We use the `populate` method of CPLEX to increase the number of solutions found; to save time we stop if the incumbent has an objective value less than -0.2. We stop the cut separation if, in the last five iterations, the relative LP bound increase was less than 0.1 percent. In case of the branch-and-cut approaches we then continue with branching. When this situation arises in the first phase of PCB we either return to the pricing loop, or we continue with the second phase in case no relevant columns have been found and more than 90% of the time limit has been reached.

Price-Cut-and-Branch We limit to five the number of columns to be added to the model per node $i \in V$ per pricing iteration. If less than 500 columns in total are added in some iteration, we increase this limit by a factor of 10. In the beginning of each pricing phase, e.g., after the cut separation rounds, the limit is reset to five again. We perform three cut separation rounds after each pricing phase. If after five consecutive pricing phases a generalized propagation constraint is not binding in the optimal LP solution it is removed again from the model. In the branch-and-cut phase we focus on finding better primal bounds by setting the `MIP emphasis` parameter in CPLEX to `feasibility`.

Table 1: Comparison of LP relaxation bounds on instance set *SW* for $\Gamma = 1$. Each row contains average values over five instances. Some LP relaxations could not be computed within 100 000 seconds (shown in parentheses next to gap value). Column #opt denotes the number of known optimal primal bounds.

			avg. LP bound [%]				avg. LP gap [%]				#opt
$\|V\|$-k	β	α	(ARC)	(COV)	$(COV)^+$	$(COV)^+_e$	(ARC)	(COV)	$(COV)^+$	$(COV)^+_e$	
50-4	0.1	0.1	0.0	5.5	98.1	100.0	100.0	95.5	22.4	21.1	5
		0.5	0.0	16.4	90.7	100.0	100.0	83.9	11.5	2.9	5
		1.0	0.9	32.4	93.6	100.0	99.1	70.0	10.8	4.5	5
	0.3	0.1	0.0	11.6	97.9	100.0	100.0	90.6	15.5	14.2	5
		0.5	0.0	21.5	82.8	100.0	100.0	83.1	32.9	19.7	5
		1.0	1.6	34.3	87.1	100.0	98.4	67.7	18.1	5.9	5
50-8	0.1	0.1	0.0	0.3	99.1	100.0	100.0	99.9	51.6	51.2	3
		0.5	0.0	0.7	81.2	100.0	100.0	99.5	43.9	29.9 (4)	1
		1.0	0.0	1.7	71.7	100.0	100.0	98.4	35.5	10.3 (4)	1
	0.3	0.1	0.0	2.2	98.2	100.0	100.0	99.0	55.0	54.2	1
		0.5	0.0	4.7	84.2	100.0	100.0	97.9	61.0	53.7 (2)	0
		1.0	0.0	5.6	67.2	100.0	100.0	96.2	53.2	30.6 (2)	0
75-4	0.1	0.1	0.0	6.5	92.4	100.0	100.0	94.3	18.0	11.2	5
		0.5	0.0	17.4	83.6	100.0	100.0	84.1	22.9	8.3	5
		1.0	0.6	38.7	90.4	100.0	99.5	61.9	10.9	1.4	5
	0.3	0.1	0.0	20.2	94.3	100.0	100.0	84.4	28.7	24.8	5
		0.5	0.0	30.6	73.6	100.0	100.0	77.6	45.6	26.5	5
		1.0	2.9	45.8	81.8	100.0	97.3	56.1	21.5	4.1	5

6.2 Computational Results

LP relaxation bounds. Table 1 compares average LP relaxation bounds $(v(M)/v((COV)^+_e)$ and average LP gaps $(UB - v(M))/UB$ to the best known primal bound UB, in percent, for additively separable and linear activation functions, i.e., $\Gamma = 1$, smaller instances from set SW, and different values of k, α, and β. Thereby, $v(M)$, where $M \in \{(ARC), (COV), (COV)^+, (COV)^+_e\}$, is the LP relaxation value of formulation M where $(COV)^+$, $(COV)^+_e$ denote the variants of (COV) augmented by heuristic and exact separation of generalized propagation constraints (2), respectively. We observe that formulation (ARC) yields extremely weak bounds (often equal to zero) and is clearly dominated by (COV). The relevance of generalized propagation constraints (2) can be clearly seen from the significantly better LP relaxation bounds obtained in variants $(COV)^+$ and $(COV)^+_e$. A comparison between the latter two also reveals that our heuristic cut separation routines work quite well in many cases, though there is potential for further improvements. The final, non-negligible gaps of $(COV)^+_e$ also indicate that the identification of further valid inequalities is a relevant topic for future research. The results from Table 1 show that the gaps (and thus the complexity of the instances) increase with increasing average node degree k. We remark, however, that this comparison might be influenced from the side effect that changing the average node degree (for a fixed number of nodes) also changes the number of arcs in the underlying graph. Higher values of α mostly lead to smaller LP gaps, which is partly due to the right-hand side lifting of inequalities (2) which can be applied more often in these cases.

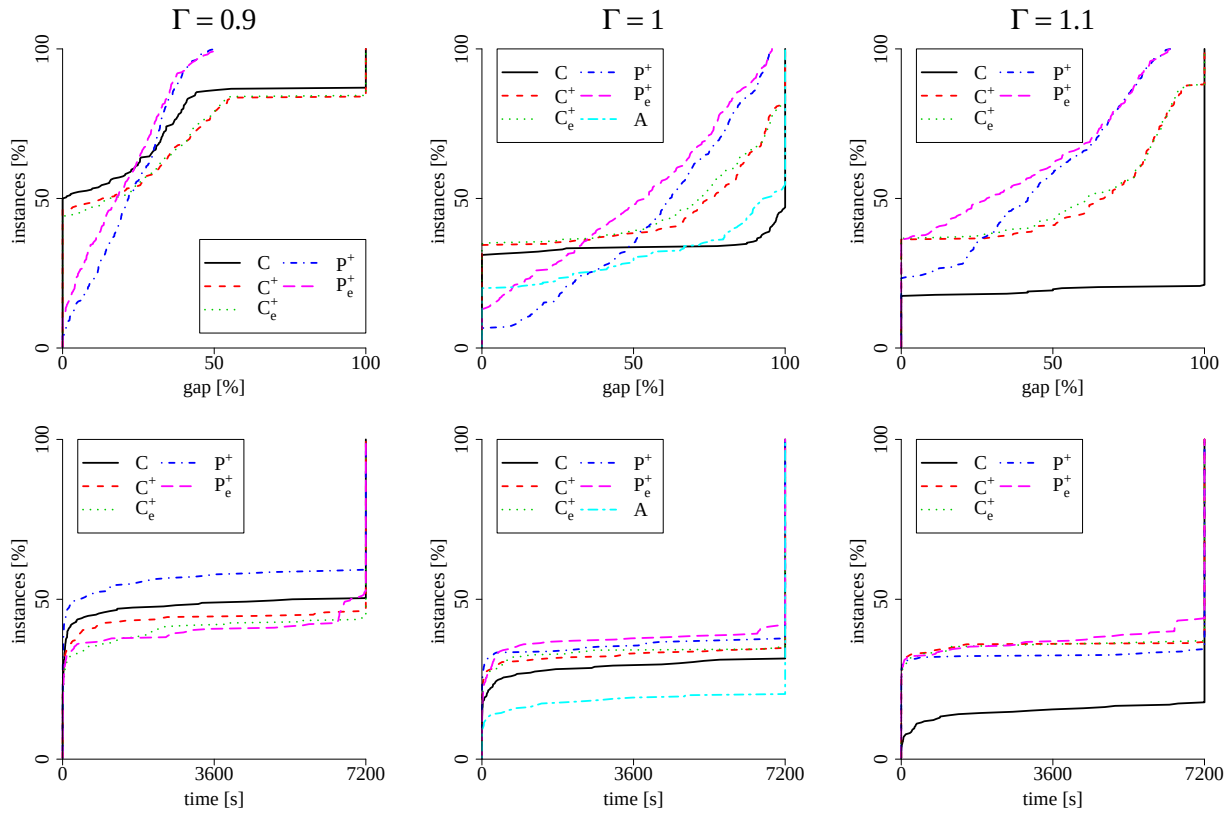

Figure 1: Cumulative relative numbers of instances for which the resulting optimality gap (the CPU-time of, respectively) is within a certain value. Results for instance set SW and $\Gamma \in \{0.9, 1, 1.1\}$.

Results for the general case. Figure 1 summarizes optimality gaps and CPU times observed for different values of Γ (i.e., including also non-linear activation functions) and all considered algorithms except G (which is only applicable in the special case discussed in the next paragraph). We observe that the required CPU times and resulting gaps (and thus the complexity for the considered methods) increase with increasing value of Γ. We also observe that, despite the fact that the exact methods based on initially enumerating all variables from (COV) perform surprisingly well, the heuristic variants P^+ and P^+_e are more reliable in the sense that the observed optimality gaps on the most challenging instances are much smaller than those of the exact methods. Concerning the different variants of C, we observe that generalized propagation constraints (2) deteriorate the performance for $\Gamma = 0.9$, but significantly improves the obtained results for the more difficult cases of $\Gamma \in \{1, 1.1\}$. Variants C^+_e and C^+ with and without exact separation overall perform very similarly. Slight advantages can be observed for variant C^+_e which seems to benefit from its stronger dual bounds. Similarly, P^+_e slightly outperforms its counterpart P^+ with heuristic cut separation. The branch-and-cut algorithm A based on model (ARC) is not only restricted to the case of additively separable and linear activation functions, but is also significantly outperformed by the other methods proposed in this article for that case. Overall, the results given in Figure 1 indicate that C^+_e and P^+_e are the two most promising variants and we will therefore focus on them in the more

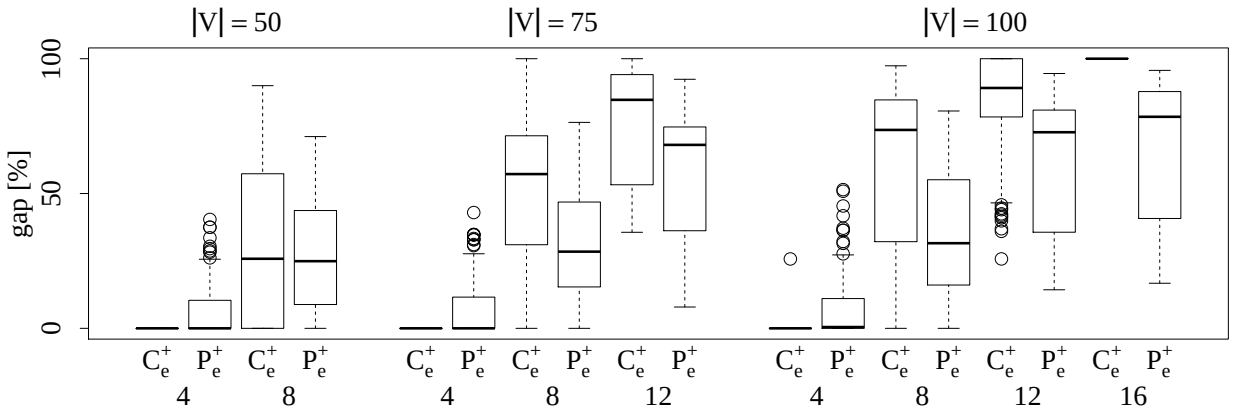

Figure 2: Distribution of optimality gaps for different average node degrees $k \in \{4, 8, 12, 16\}$ and numbers of nodes $|V|$ for instances from set SW.

fine-grained results given in Figure 2 that compare the distribution of final optimality gaps by numbers of nodes and average node degrees. These results clearly confirm the conjecture that the average node degree is a major factor of the complexity of an instance. On the contrary, it seems that the impact of an increasing number of nodes is not too large as long as the total number of arcs remains constant (e.g., instances with $|V| = 100$ and $k = 4$ are easier than those with $|V| = 50$ and $k = 8$), thus suggesting a reasonable scalability of our approaches for sparse graphs. These results also show that the heuristic, column-generation based algorithm P_e^+ consistently outperforms the exact method C_e^+ if the average node degree is larger than four. Finally, the detailed results given in the appendix in Tables 3 to 5 suggest that instances with higher values of α are harder to solve to optimality, even though the dual bounds are better in these cases.

Results for special case considered in Günneç et al. [13]. Table 2 summarizes our results for the special case of the GLCIP with additively separable and linear activation functions, equal influence from all neighbors, $\alpha = 1$, and continuous incentives whose costs are equal to the corresponding hurdle reduction. Results for C_e^+ and P_e^+ are not reported, since the exact separation of propagation inequalities is not competitive because of the large graph sizes. We first observe that this case seems significantly easier to solve. The maximum size of instances that can be solved to proven optimality is orders of magnitude larger than for the general case, and the optimality gaps are comparably small. We also conclude that our exact and heuristic algorithms with heuristic cut separation, i.e., C^+ and P^+, clearly outperform the previous state-of-the-art approach G from Günneç et al. [13] in most cases, despite the fact that they are not specialized or tuned to this particular case.

7 Conclusions

We have introduced and studied a new optimization problem motivated from viral marketing in social networks that generalizes many previously considered problem variants. An ILP formulation with an exponential number of variables that allows to consider arbitrary activation functions, as well as strengthening valid inequalities, exact and heuristic algorithms based on this formulation

Table 2: Computational results on instance set GRZ. Dashes indicate no available gaps or reached time limits.

	avg. gap in %					avg. time in sec.				
$\|V\|$-k	G	A	C	C^+	P^+	G	A	C	C^+	P^+
1000-4	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	6	169	**1**	**1**	7
10000-4	0.2	1.9	**0.0**	**0.0**	**0.0**	-	-	53	**52**	773
50000-4	0.4	3.0	**0.0**	**0.0**	**0.0**	-	-	**5630**	6929	6857
100000-4	0.2	3.3	2.8	**0.1**	**0.1**	-	-	-	-	**7144**
5000-8	6.4	6.8	5.2	2.1	**2.0**	-	-	-	-	-
2500-16	**12.9**	100.0	-	-	53.2	-	-	-	-	-

have been introduced. Computational results show that our approaches significantly outperform (extensions of) existing algorithms on special cases of the general problem. Promising directions for future work include the development of an improved branch-price-and-cut algorithm, by studying the integration of stabilization techniques, improving the proposed heuristic cut separation routines, and identifying further valid inequalities.

References

[1] E. Ackerman, O. Ben-Zwi, and G. Wolfovitz. Combinatorial model and bounds for target set selection. *Theoretical Computer Science*, 411(44):4017–4022, 2010.

[2] O. Ben-Zwi, D. Hermelin, D. Lokshtanov, and I. Newman. Treewidth governs the complexity of target set selection. *Discrete Optimization*, 8(1):87–96, 2011.

[3] S. P. Borgatti. Identifying sets of key players in a social network. *Computational & Mathematical Organization Theory*, 12(1):21–34, 2006.

[4] N. Chen. On the approximability of influence in social networks. *SIAM Journal on Discrete Mathematics*, 23(3):1400–1415, 2009.

[5] W. Chen, L. V. Lakshmanan, and C. Castillo. Information and influence propagation in social networks. *Synthesis Lectures on Data Management*, 5(4):1–177, 2013.

[6] P. Domingos and M. Richardson. Mining the network value of customers. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 57–66. ACM, 2001.

[7] P. A. Dreyer, Jr. and F. S. Roberts. Irreversible k-threshold processes: Graph-theoretical threshold models of the spread of disease and of opinion. *Discrete Applied Mathematics*, 157 (7):1615–1627, 2009.

[8] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, New York, 1979.

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

[10] M. Grötschel, M. Jünger, and G. Reinelt. On the acyclic subgraph polytope. *Mathematical Programming*, 33(1):28–42, 1985.

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

[12] D. Günneç. *Integrating Social Network Effects in Product Design*. PhD thesis, University of Maryland, 2012.

[13] D. Günneç, S. Raghavan, and R. Zhang. Tailored incentives and least cost influence maximization on social networks. Technical report, University of Maryland, 2017.

[14] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[15] D. Kempe, J. Kleinberg, and É. Tardos. Influential nodes in a diffusion model for social networks. In *Proceedings of the 32nd International Conference on Automata, Languages and Programming*, pages 1127–1138. Springer, 2005.

[16] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. *Theory of Computing*, 11(4):105–147, 2015.

[17] M. Kimura, K. Saito, and R. Nakano. Extracting influential nodes for information diffusion on a social network. In *AAAI*, volume 7, pages 1371–1376, 2007.

[18] J. Leskovec, L. A. Adamic, and B. A. Huberman. The dynamics of viral marketing. *ACM Trans. Web*, 1(1):1–39, 2007.

[19] S. Raghavan and R. Zhang. Weighted target set selection on social networks. Technical report, University of Maryland, 2016.

[20] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 61–70. ACM, 2002.

[21] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world'-networks. *Nature*, 393 (6684):440–442, 1998.

[22] H.-H. Wu and S. Küçükyavuz. A two-stage stochastic programming approach for influence maximization in social networks. *Computational Optimization and Applications*, 2017. online first.

A Detailed Results

Table 3: Results on SW instances for $\Gamma = 0.9$. Note that P is a heuristic. Dashes indicate no available gaps / set variables or reached time limits.

			avg. gap [%]					avg. time [s]					avg. # set variables		
$\|V\|$-k	β	α	C	C^+	C^+_e	P^+	P^+_e	C	C^+	C^+_e	P^+	P^+_e	C	P^+	P^+_e
50-4	0.1	0.1	**0.0**	**0.0**	**0.0**	21.4	17.9	**0**	**0**	3	**0**	2	883	315	319
		0.5	**0.0**	**0.0**	**0.0**	11.6	6.3	**1**	2	26	**1**	27	883	344	353
		1.0	**0.0**	**0.0**	**0.0**	1.1	0.2	20	**1**	2	**1**	**1**	883	340	343
	0.3	0.1	**0.0**	**0.0**	**0.0**	17.3	16.0	**0**	1	10	**0**	3	1383	327	333
		0.5	**0.0**	**0.0**	**0.0**	12.1	7.8	**2**	7	44	**2**	15	1383	374	393
		1.0	**0.0**	**0.0**	**0.0**	3.3	1.1	86	**7**	21	21	9	1383	384	396
50-8	0.1	0.1	**0.0**	**0.0**	**0.0**	35.2	33.2	17	64	1007	**3**	435	17932	1303	1425
		0.5	**0.0**	**0.0**	2.3	19.2	12.5	80	2076	3879	**57**	6560	17932	1525	1839
		1.0	22.5	7.8	2.2	7.2	**1.5**	-	7171	4105	-	**3700**	17932	1901	2182
	0.3	0.1	**0.0**	**0.0**	**0.0**	34.9	30.0	27	117	1005	**6**	388	39809	1502	1712
		0.5	**0.0**	15.9	20.5	27.4	21.9	2116	6692	-	**565**	4852	39809	1827	2074
		1.0	22.0	25.3	20.2	13.1	**9.1**	-	-	**6249**	-	7191	39809	2280	2471
75-4	0.1	0.1	**0.0**	**0.0**	**0.0**	22.1	16.8	1	1	4	**0**	3	1357	491	493
		0.5	**0.0**	**0.0**	**0.0**	6.3	3.8	**1**	**1**	11	**1**	30	1357	509	532
		1.0	**0.0**	**0.0**	**0.0**	0.9	0.3	236	3	2	**1**	6	1357	514	533
	0.3	0.1	**0.0**	**0.0**	**0.0**	15.1	12.3	**1**	2	17	**1**	19	1808	515	528
		0.5	**0.0**	**0.0**	**0.0**	13.6	7.9	**4**	37	168	6	65	1808	546	594
		1.0	**0.0**	**0.0**	**0.0**	2.2	0.3	306	14	14	**7**	14	1808	553	575
75-8	0.1	0.1	**0.0**	**0.0**	7.9	35.0	32.1	45	767	3870	**16**	5330	29557	2140	2459
		0.5	**0.0**	12.6	13.1	16.8	10.9	1351	-	6890	**332**	6887	29557	2558	2839
		1.0	30.7	31.4	26.0	10.4	**4.1**	-	-	-	-	-	29557	3354	3761
	0.3	0.1	**0.0**	**0.0**	4.7	32.7	28.2	66	544	5078	**25**	3575	52163	2453	2760
		0.5	**13.8**	32.8	29.9	29.2	25.6	-	-	-	**6192**	-	52163	2596	2977
		1.0	32.4	32.4	30.0	20.1	**12.5**	-	-	-	-	-	52163	3870	4285
75-12	0.1	0.1	**17.9**	49.9	50.0	41.9	42.1	6852	-	-	**479**	7160	527510	7437	10050
		0.5	32.9	45.4	43.5	**30.9**	31.1	-	-	-	-	-	527510	12136	12888
		1.0	40.7	39.4	39.2	19.7	**12.4**	-	-	-	-	-	310536	13207	13870
	0.3	0.1	40.0	61.5	61.6	**32.3**	35.1	-	-	-	**1314**	-	1464607	11300	13214
		0.5	52.8	58.0	57.9	38.0	**36.4**	-	-	-	-	-	1464607	12864	13837
		1.0	52.0	50.7	50.6	25.7	**22.8**	-	-	-	-	-	1464607	12227	12960
100-4	0.1	0.1	**0.0**	**0.0**	**0.0**	13.5	9.8	**1**	2	10	**1**	4	1826	627	630
		0.5	**0.0**	**0.0**	**0.0**	6.6	3.6	3	8	42	**2**	85	1826	668	689
		1.0	**0.0**	**0.0**	**0.0**	0.9	**0.0**	1310	11	12	13	**10**	1826	747	745
	0.3	0.1	**0.0**	**0.0**	**0.0**	13.0	10.4	1	1	9	**0**	4	2597	645	651
		0.5	**0.0**	**0.0**	**0.0**	10.4	6.1	11	73	456	**5**	71	2597	752	792
		1.0	1.0	**0.0**	**0.0**	1.8	0.6	4444	165	**47**	419	55	2597	795	807
100-8	0.1	0.1	**0.0**	0.7	7.2	27.4	22.2	83	3992	5294	**33**	6599	38820	2874	3359
		0.5	**4.2**	23.7	28.2	16.6	13.4	5314	-	-	**3307**	7182	38820	3274	3848
		1.0	30.3	34.2	32.2	12.6	**3.9**	-	-	-	-	-	38820	4403	4523
	0.3	0.1	**0.0**	1.4	10.5	27.7	23.5	130	2761	6181	**75**	6718	73397	3537	4228
		0.5	**15.1**	34.6	36.9	25.4	22.8	-	-	-	**6194**	-	73397	3872	4559
		1.0	26.2	27.6	27.1	17.3	**10.9**	-	-	-	-	-	73397	5553	6172
100-12	0.1	0.1	**36.0**	53.7	52.6	42.6	45.2	-	-	-	**2917**	-	684855	9829	12588
		0.5	34.6	44.7	42.3	33.5	**30.7**	-	-	-	-	-	684855	15312	16452
		1.0	38.9	38.0	41.5	24.3	**18.8**	-	-	-	-	-	125694	15455	17800
	0.3	0.1	38.9	47.7	47.7	35.9	**35.1**	-	-	-	**4200**	-	1308151	15225	17345
		0.5	42.3	46.2	46.1	35.9	**35.2**	-	-	-	-	-	1308151	16804	17835
		1.0	37.0	37.6	37.6	26.4	**25.9**	-	-	-	-	-	1032888	14349	15369
100-16	0.1	0.1	-	-	-	43.4	**41.0**	-	-	-	-	-	-	33666	34631
		0.5	-	-	-	33.8	**33.6**	-	-	-	-	-	-	38008	38738
		1.0	-	-	-	28.7	**27.2**	-	-	-	-	-	-	16420	18538
	0.3	0.1	-	-	-	42.6	**42.3**	-	-	-	-	-	-	39312	39597
		0.5	-	-	-	**35.6**	**35.6**	-	-	-	-	-	-	34741	35136
		1.0	-	-	-	26.0	**25.5**	-	-	-	-	-	-	13215	13009

Table 4: Results on SW instances for $\Gamma = 1$. Note that P is a heuristic. Dashes indicate no available gaps / set variables or reached time limits.

			avg. gap [%]						avg. time [s]						avg. # set variables		
$\|V\|$-k	β	α	A	C	C^+	C^+_e	P^+	P^+_e	A	C	C^+	C^+_e	P^+	P^+_e	C	P^+	P^+_e
50-4	0.1	0.1	10.0	**0.0**	**0.0**	**0.0**	22.9	21.1	1444	2	1	3	**0**	2	783	248	249
		0.5	10.0	**0.0**	**0.0**	**0.0**	10.8	2.9	1832	70	5	5	**1**	11	783	259	256
		1.0	17.3	**0.0**	**0.0**	**0.0**	10.1	4.5	2973	99	2	2	**0**	1	783	303	303
	0.3	0.1	**0.0**	**0.0**	**0.0**	**0.0**	15.4	14.2	1	**0**	**0**	1	**0**	2	1171	263	261
		0.5	**0.0**	**0.0**	**0.0**	**0.0**	33.0	19.7	1087	1040	92	274	**5**	29	1171	312	317
		1.0	5.8	**0.0**	**0.0**	**0.0**	20.0	5.9	2585	945	19	29	21	**8**	1171	355	358
50-8	0.1	0.1	83.6	98.2	**17.3**	25.9	53.0	52.4	-	-	5103	4928	3807	**299**	15655	719	749
		0.5	89.6	99.2	45.0	41.8	46.3	**34.4**	-	-	6571	5872	6820	**5812**	15655	1511	1657
		1.0	90.6	97.8	66.0	25.7	39.4	**12.5**	-	-	-	**5894**	-	6348	15655	1476	1816
	0.3	0.1	62.9	89.5	**28.9**	39.5	55.1	54.3	-	-	7083	7123	1329	**280**	35360	862	990
		0.5	90.8	96.3	74.5	66.2	63.2	**59.8**	-	-	-	-	-	-	35360	1859	2013
		1.0	85.1	95.2	67.5	61.6	61.3	**40.4**	-	-	-	-	-	-	35360	1973	2165
75-4	0.1	0.1	10.0	**0.0**	**0.0**	**0.0**	23.1	23.6	1454	13	**1**	6	**1**	8	1192	414	416
		0.5	20.0	**0.0**	**0.0**	**0.0**	22.8	8.3	4887	294	27	133	**5**	144	1192	405	420
		1.0	21.6	**0.0**	**0.0**	**0.0**	7.2	1.4	5679	1350	5	5	**2**	**2**	1192	474	450
	0.3	0.1	**0.0**	**0.0**	**0.0**	**0.0**	29.3	24.8	2	1	2	11	**0**	16	1574	423	425
		0.5	8.2	**0.0**	**0.0**	**0.0**	42.4	26.2	2717	1158	348	1038	**23**	457	1574	468	496
		1.0	22.2	5.6	**0.0**	**0.0**	20.5	4.0	5988	1618	697	**57**	758	102	1574	504	530
75-8	0.1	0.1	93.2	99.7	67.4	70.9	62.4	**62.0**	-	-	-	-	-	**6640**	25397	1099	1299
		0.5	95.4	99.0	78.1	65.3	62.0	**45.3**	-	-	-	-	-	-	25397	2415	2794
		1.0	95.6	96.6	80.8	65.4	58.7	**30.0**	-	-	-	-	-	-	25397	2337	2947
	0.3	0.1	89.7	94.1	65.2	69.5	59.0	**57.6**	-	-	-	-	**5345**	6159	45887	1330	1749
		0.5	96.3	97.6	86.4	85.6	71.0	**68.5**	-	-	-	-	-	-	45887	3209	2993
		1.0	94.3	97.9	82.5	70.5	62.1	**40.2**	-	-	-	-	-	-	45887	3395	3189
75-12	0.1	0.1	100.0	100.0	84.5	84.4	**71.4**	**71.4**	-	-	-	-	-	-	469196	2158	2568
		0.5	100.0	100.0	93.2	94.2	81.9	**70.5**	-	-	-	-	-	-	469196	5284	12487
		1.0	100.0	100.0	94.1	93.9	83.6	**79.3**	-	-	-	-	-	-	469196	6050	8356
	0.3	0.1	100.0	100.0	87.4	87.4	73.5	**73.3**	-	-	-	-	-	-	1342667	2836	6163
		0.5	100.0	100.0	98.4	97.5	90.1	**85.1**	-	-	-	-	-	-	1342667	5362	11652
		1.0	100.0	100.0	98.2	98.2	91.0	**86.2**	-	-	-	-	-	-	1342667	7115	10240
100-4	0.1	0.1	**0.0**	**0.0**	**0.0**	**0.0**	25.8	20.8	277	3	6	9	**1**	47	1620	542	543
		0.5	44.6	**0.0**	**0.0**	**0.0**	34.0	5.3	-	703	281	266	**12**	380	1620	523	542
		1.0	57.5	3.7	**0.0**	**0.0**	12.8	0.6	-	3301	66	**13**	61	27	1620	629	629
	0.3	0.1	**0.0**	**0.0**	**0.0**	**0.0**	37.0	32.3	13	3	6	208	**2**	126	2271	557	565
		0.5	26.3	**4.4**	5.7	5.1	50.9	27.1	5904	2062	3277	3460	**1346**	1537	2271	645	682
		1.0	50.1	8.9	**0.0**	**0.0**	20.9	3.9	-	5160	1835	**229**	1535	361	2271	726	744
100-8	0.1	0.1	86.0	93.6	71.7	74.9	63.8	**62.0**	-	-	-	-	-	**6998**	33917	1645	1914
		0.5	93.8	98.1	88.2	80.9	60.4	**34.6**	-	-	-	-	-	**7067**	33917	3094	3806
		1.0	93.3	96.5	90.5	86.8	67.5	**29.2**	-	-	-	-	-	-	33917	3076	3941
	0.3	0.1	97.1	100.0	77.2	81.2	64.5	**64.1**	-	-	-	-	-	**7172**	65758	1808	2236
		0.5	99.1	100.0	92.8	93.9	84.0	**75.4**	-	-	-	-	-	-	65758	4796	4515
		1.0	98.4	98.8	93.0	89.5	81.0	**55.9**	-	-	-	-	-	-	65758	5730	4720
100-12	0.1	0.1	100.0	100.0	87.4	87.7	77.1	**76.8**	-	-	-	-	-	-	616486	3569	5238
		0.5	100.0	100.0	93.9	94.6	85.9	**78.7**	-	-	-	-	-	-	616486	5254	11268
		1.0	100.0	100.0	94.0	94.2	**85.9**	86.0	-	-	-	-	-	-	616486	5664	5701
	0.3	0.1	100.0	100.0	100.0	100.0	78.1	**77.9**	-	-	-	-	-	-	1192315	3919	9691
		0.5	100.0	100.0	100.0	100.0	93.2	**90.7**	-	-	-	-	-	-	1192315	4920	12642
		1.0	100.0	100.0	99.5	100.0	**93.8**	93.9	-	-	-	-	-	-	1192315	7213	9452
100-16	0.1	0.1	100.0	-	-	-	79.8	**79.1**	-	-	-	-	-	-	-	3710	5465
		0.5	100.0	-	-	-	91.3	**89.5**	-	-	-	-	-	-	-	6574	17380
		1.0	100.0	-	-	-	92.2	**91.0**	-	-	-	-	-	-	-	9082	8144
	0.3	0.1	100.0	-	-	-	**80.7**	81.4	-	-	-	-	-	-	-	7084	14643
		0.5	100.0	-	-	-	**94.5**	95.0	-	-	-	-	-	-	-	7404	36543
		1.0	100.0	-	-	-	**94.8**	95.0	-	-	-	-	-	-	-	10003	10543

Table 5: Results on SW instances for $\Gamma = 1.1$. Note that P is a heuristic. Dashes indicate no available gaps / set variables or reached time limits.

			avg. gap [%]					avg. time [s]					avg. # set variables		
$\|V\|$-k	β	α	C	C^+	C^+_e	P^+	P^+_e	C	C^+	C^+_e	P^+	P^+_e	C	P^+	P^+_e
50-4	0.1	0.1	20.0	**0.0**	**0.0**	**0.0**	**0.0**	1507	**0**	**0**	**0**	**0**	664	187	195
		0.5	8.3	**0.0**	**0.0**	**0.0**	**0.0**	1730	**0**	2	**0**	**0**	664	215	215
		1.0	8.3	**0.0**	**0.0**	**0.0**	**0.0**	1497	**0**	**0**	**0**	**0**	664	224	216
	0.3	0.1	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	123	**0**	**0**	**0**	**0**	920	210	208
		0.5	10.0	**0.0**	**0.0**	11.1	**0.0**	3547	5	**4**	**4**	**4**	920	202	203
		1.0	13.0	**0.0**	**0.0**	12.9	3.6	3360	2	5	**1**	6	920	236	242
50-8	0.1	0.1	100.0	**16.5**	20.8	33.9	33.2	-	**3411**	4530	5833	6890	10958	406	454
		0.5	100.0	34.8	25.8	20.3	**7.1**	-	5997	5235	4328	**4202**	10958	764	1283
		1.0	100.0	30.9	30.5	31.7	**6.9**	-	4711	4649	6757	**3937**	10958	687	1252
	0.3	0.1	100.0	38.7	**33.7**	43.5	43.5	-	6071	5469	**4614**	5904	22739	484	509
		0.5	100.0	69.0	61.6	44.1	**16.8**	-	-	-	-	**5666**	22739	924	2053
		1.0	100.0	72.9	72.5	38.0	**12.5**	-	-	-	-	**5782**	22739	927	2098
75-4	0.1	0.1	40.0	**0.0**	**0.0**	**0.0**	**0.0**	4289	**0**	**0**	**0**	**0**	1007	307	313
		0.5	60.0	**0.0**	**0.0**	**0.0**	**0.0**	5349	**0**	1	**0**	**0**	1007	303	287
		1.0	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	1996	**0**	**0**	**0**	**0**	1007	328	330
	0.3	0.1	20.0	**0.0**	**0.0**	7.8	7.0	1550	**2**	34	4	3	1282	372	370
		0.5	32.2	**0.0**	**0.0**	19.0	4.7	4487	**162**	488	1442	249	1282	360	377
		1.0	30.6	**0.0**	**0.0**	16.2	4.6	3291	150	**88**	2880	678	1282	356	371
75-8	0.1	0.1	100.0	52.6	54.0	39.2	**36.8**	-	-	-	-	-	17310	718	713
		0.5	100.0	66.9	48.7	27.9	**11.8**	-	-	-	-	**6205**	17310	860	1737
		1.0	100.0	63.3	54.6	26.0	**11.2**	-	-	-	-	**6530**	17310	923	1741
	0.3	0.1	100.0	53.4	59.1	36.0	**34.0**	-	-	-	6947	**6555**	29829	784	968
		0.5	100.0	74.2	77.8	40.0	**29.4**	-	-	-	-	**6057**	29829	1641	2375
		1.0	100.0	76.5	76.1	42.3	**30.3**	-	-	-	-	**6190**	29829	1489	2548
75-12	0.1	0.1	100.0	79.9	79.3	**60.5**	63.1	-	-	-	-	-	269538	1749	1773
		0.5	100.0	85.5	85.2	65.8	**64.0**	-	-	-	-	-	269538	4493	4924
		1.0	100.0	84.9	82.2	63.6	**59.3**	-	-	-	-	-	269538	3794	4215
	0.3	0.1	100.0	81.4	81.1	**68.9**	70.3	-	-	-	-	-	711655	2362	2714
		0.5	100.0	88.9	87.8	76.7	**75.9**	-	-	-	-	-	711655	5164	5624
		1.0	100.0	89.5	89.6	**74.8**	75.3	-	-	-	-	-	711655	5267	4838
100-4	0.1	0.1	80.0	**0.0**	**0.0**	**0.0**	**0.0**	5760	**0**	1	**0**	**0**	1363	405	428
		0.5	80.0	**0.0**	**0.0**	9.3	**0.0**	5772	5	7	**2**	22	1363	388	382
		1.0	80.0	**0.0**	**0.0**	9.8	**0.0**	5761	2	6	**1**	2	1363	472	470
	0.3	0.1	100.0	**0.0**	**0.0**	16.7	14.7	-	**18**	224	1345	1055	1839	471	473
		0.5	100.0	**0.0**	**0.0**	29.9	**0.0**	-	**84**	176	2943	116	1839	473	518
		1.0	86.7	**0.0**	**0.0**	16.2	3.3	-	**81**	110	2573	2838	1839	510	527
100-8	0.1	0.1	100.0	65.6	66.2	40.8	**40.0**	-	-	-	-	-	23428	1044	1050
		0.5	100.0	58.3	47.8	36.4	**11.7**	-	7117	-	-	**5237**	23428	1372	1796
		1.0	100.0	73.5	64.6	35.8	**19.2**	-	-	-	-	**5644**	23428	1228	1818
	0.3	0.1	100.0	72.6	79.8	**49.7**	51.6	-	-	-	-	-	43209	1561	1580
		0.5	100.0	85.1	83.5	60.7	**45.5**	-	-	-	-	-	43209	2522	2751
		1.0	100.0	89.9	85.8	55.1	**48.8**	-	-	-	-	-	43209	2566	2708
100-12	0.1	0.1	100.0	82.7	82.3	**64.1**	65.1	-	-	-	-	-	362885	1765	2155
		0.5	100.0	84.6	83.5	**66.8**	68.7	-	-	-	-	-	362885	4449	4393
		1.0	100.0	83.6	85.2	68.7	**66.4**	-	-	-	-	-	362885	4273	4027
	0.3	0.1	100.0	87.3	87.3	73.7	**73.6**	-	-	-	-	-	1184188	2442	3201
		0.5	100.0	93.6	93.3	**80.7**	81.3	-	-	-	-	-	1184188	4251	4665
		1.0	100.0	93.3	92.2	**80.9**	81.3	-	-	-	-	-	1184188	4495	4131
100-16	0.1	0.1	100.0	100.0	100.0	**72.5**	73.8	-	-	-	-	-	1476060	2393	2586
		0.5	100.0	100.0	100.0	79.1	**76.9**	-	-	-	-	-	1476060	6008	6632
		1.0	100.0	100.0	100.0	78.4	**78.0**	-	-	-	-	-	1476060	6275	5977
	0.3	0.1	-	-	-	**77.2**	77.8	-	-	-	-	-	-	2395	4294
		0.5	-	-	-	86.1	**85.6**	-	-	-	-	-	-	6894	7795
		1.0	-	-	-	**85.9**	86.1	-	-	-	-	-	-	6876	6712