Computers & Operations Research 135 (2021) 105441

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor

Generalizations, formulations and subgradient based heuristic with dynamic programming procedure for target set selection problems

Santiago V. Ravelo [a,*], Cláudio N. Meneses [b]

[a] Institute of Informatics, Federal University of Rio Grande do Sul, Porto Alegre, Brazil
[b] Center of Mathematics, Computation and Cognition, Federal University of ABC, São Paulo, Brazil

ARTICLE INFO

Keywords:
Target set selection
Integer linear programming
Subgradient method
Lagrangian relaxation
Dynamic programming
Heuristic solution

ABSTRACT

Target set selection problems model influence processes through networks. Usually, these problems seek a set of individuals to be initially influenced by some content and will help to propagate the content to other individuals of the network. This work considers generalizations for the minimization and maximization versions of these problems. We propose binary linear models and Lagrangian relaxations that, for general cases, offer strictly better bounds than the linear relaxations. To efficiently solve our relaxations we design dynamic programming algorithms and embedded them in subgradient methods to find the best possible bounds. We also use the subgradient methods to construct feasible heuristic solutions and, to assess the effectiveness of our approaches, we run computational experiments on instances generated from real-world datasets. The experiments indicate that our heuristics obtain almost optimal solutions with good bounds in a very short time and, in some cases, they guarantee optimality for the constructed solutions.

1. Introduction

Target Set Selection problem (TSS) arises from a diverse range of areas including marketing, politics, social media and bioinformatics. Such problems receive a network with influence relations between each pair of individuals, allowing to model the spread of products, ideas, news or contagious diseases. Aiming to represent a product acquisition (the adoption of an idea, a news belief or a disease infection) by an individual, it is said the individual is influenced (or activated) and every activated individual becomes an influencer, contributing to propagate the products (ideas, news or diseases), by influencing (or activating) other individuals of the network.

A solution for the TSS is a set of individuals to be initially activated, while the objective may change depending on the version. The most common versions are: to minimize the size of the initial set of individuals while guaranteeing to activate the whole network or part of it (Min-TSS), and to maximize the part of the network that will be activated while guaranteeing the size of the initially activated individuals is not greater than a given upper bound (Max-TSS).

Regardless of the problem version, solving the TSS may help companies to increase the sales of products with low marketing costs. Also, solutions for those problems could mitigate the disinformation created by "fake news" by selecting individuals that will spread accurate news and "good practices" for information consumption, with positive impacts in politics, public opinion and population beliefs (Lazer et al., 2018). Those solutions may even help to model the extension of contagious disease development from a small group of initially infected members in a community (Dreyer and Roberts, 2009).

The first formalizing the TSS as a combinatorial problem were Kempe et al. (2003), where a polynomial-time approximation algorithm and an NP-hardness proof were given for a probabilistic TSS inspired in the work of Richardson and Domingos (2002). Since then, many studies were done on the different variants of the TSS. NP-hard proofs for the Min-TSS and the Max-TSS were given by several authors (Centeno et al., 2011; Chen, 2009; Cicalese et al., 2015; Dreyer and Roberts, 2009; Narayanan and Wu, 2020), followed by approximation and tractability results for different network structures such as paths, cycles, trees among other graphs (Chiang et al., 2013; Nichterlein et al., 2013), mathematical formulations (Ackerman et al., 2010; Baghbani et al., 2019; Fardad and Kearney, 2017), heuristic algorithms (Cordasco et al., 2018) and enumerative algorithms as Branch & Cut (Raghavan and Zhang, 2019). Also, there were studies combining the minimization and maximization objectives, where exact

* Corresponding author.
E-mail addresses: santiago.ravelo@inf.ufrgs.br (S.V. Ravelo), claudio.meneses@ufabc.edu.br (C.N. Meneses).

https://doi.org/10.1016/j.cor.2021.105441
Received 22 July 2020; Received in revised form 14 June 2021; Accepted 15 June 2021
Available online 19 June 2021

algorithms were proposed (Ben-Zwi et al., 2011; Bliznets and Sagunov, 2019) or evolutionary metaheuristics were designed considering some generalizations for practical applicability (Ravelo et al., 2020).

Despite the several results and approaches to the TSS and its variants, to the best of our knowledge, no proposals are using Lagrangian relaxation techniques to obtain bounds or to construct solutions for some exact nor heuristic method. Therefore, inspired by recent and successful applications of the Lagrangian relaxation to solve other problems (Gaudioso et al., 2017; Hernández-Leandro et al., 2019; Bulbul and Kasimbeyli, 2021; Weiner et al., 2021; Wu et al., 2021), we study the application of Lagrangian relaxations for different variants of TSS. Furthermore, in this work, we propose binary linear programs for generalized versions of minimum and maximum TSS with Lagrangian relaxations that, in general, provide better bounds than those produced by the linear programming relaxation. We also design pseudo-polynomial time algorithms based on dynamic programming techniques to solve the Lagrangian relaxations and we use subgradient methods to obtain the Lagrangian relaxations bounds and to construct feasible heuristic solutions. We test our algorithms over instances created from real-world datasets, comparing the quality of our solutions with optimal values (when found), while our bounds are compared with the ones obtained by linear relaxations. We also execute tests by embedding our subgradient method in a Branch & Bound algorithm to exactly solve the problem instances.

The rest of the paper is organized as follows. In Section 2 we give some basic notations and formally define the problems we study and in Section 3 we present new binary linear programming formulations for the problems. In Section 4 we propose Lagrangian relaxations for the mathematical formulations and show how exactly solve the relaxations with dynamic programming techniques. We also prove that the bounds obtained by our Lagrangian relaxation are better than those calculated by a linear relaxation in general scenarios and, in Section 5 we propose a subgradient based method to obtain those bounds and also to construct feasible heuristic solutions. In Section 6, we execute experiments over instances constructed from real-world datasets to test our subgradient method independently and as part of a Branch & Bound algorithm. Finally, in Section 7, the conclusions and future directions are given.

2. Definitions and notations

This paper considers some of the generalizations given by Ravelo et al. (2020) for the minimization and maximization Target Set Selection problems. These problems receive as input a network of individuals that, at any moment, can be in one of the two states, activated and non-activated, but once an individual is activated, he/she remains in that state (i.e., activated individuals never change to non-activated state). Also, every individual has a rational activation influence value in $[0,1]$ over each one of the other individuals in the network, so a non-activated individual changes its state to activated iff the sum of the influences of the activated individuals over him is at least one. More precisely, we define the situation in which an individual will change its activation state depending on the influences over him as:

Definition 1. Given a finite non-empty set I of individuals, a rational influence function $\psi : I \times I \to [0,1]$ over all pairs of individuals, a set $A \subset I$ of all activated individuals in I and a non-activated individual $i \in I \setminus A$, we say that i **is influenced by** A iff $\sum_{j \in A} \psi(j,i) \geq 1$.

If we activate all individuals influenced by a given set of activated ones, then we may define the concept of influence propagation step:

Definition 2. Let I be a finite non-empty set of individuals, $\psi : I \times I \to [0,1]$ a rational influence function over all pairs of I and a set $A \subseteq I$ of all activated individuals in I. If A' is the set of all individuals in I influenced by A and we activate all individuals in A', then the **influence propagation step from** A is the set $\rho(A) = A' \cup A$ with all activated individuals of I.

The influence propagation process from a set of initially activated individuals consists of activating all non-activated individuals influenced by the activated ones, repeating this process until no new individual can be activated. Formally:

Definition 3. Given a finite non-empty set I of individuals, a rational influence function $\psi : I \times I \to [0,1]$ over all pairs of I and a set $A \subseteq I$ with all initially activated individuals of I. For any integer $t \geq 1$, we define recursively the **influence propagation** t **steps from** A as the set $\rho^t(A) = \rho(\rho^{t-1}(A))$, resulting from activating all individuals influenced by the active set of individuals $\rho^{t-1}(A)$, where $\rho^0(A) = A$ and $\rho^1(A) = \rho(A)$. Considering $T \geq 0$ is the first integer for which $\rho^T(A) = \rho(\rho^T(A))$, we define the set $\rho^T(A) \subseteq I$ as the **result of the influence propagation process from** A.

Usually, the objective of the Min-TSS is to find a set of initially activated individuals A with minimum cardinality, that guarantees $|\rho^T(A)|$ is greater than or equal to a given number of individuals (Centeno et al., 2011; Dreyer and Roberts, 2009). For the Max-TSS the objective is also to find a set of initially activated individuals A, but in this version $\rho^T(A)$ must have maximum cardinality, while $|A|$ cannot be greater than a given number of individuals (Ackerman et al., 2010; Baghbani et al., 2019; Cicalese et al., 2015; Narayanan and Wu, 2020).

A generalization for both problems is the weighted TSS (Cicalese et al., 2015; Raghavan and Zhang, 2019; Ravelo et al., 2020), where in order to initially activate the individuals, an effort is required (which may be different for different individuals). So, the objective of the minimization version of the problem is updated to minimize the sum of the efforts to activate the initial set A, while the maximization version must ensure that the sum of the efforts to activate the initial set A is at most a given value.

In practical applications, not only different individuals require different efforts to be activated, but they also give different rewards if activated (Ravelo et al., 2020). For that reason, in this work, we generalize the weighted TSS and consider that each individual, if activated, will give a reward value. Then, in the minimization version of the problem the sum of the individuals rewards in $\rho^T(A)$ should be at least some given value, while the objective of the maximization version is updated to maximize the sum of the individuals rewards in $\rho^T(A)$.

Formally, we define the generalized minimization version of the Min-TSS as:

Problem 1. Minimum weighted Effort–Reward Target Set Selection problem (Min-ER-TSS)

Input: A tuple $\langle I, \psi, \alpha, \beta, K \rangle$, where:

- I, finite non-empty set of individuals,
- $\psi : I \times I \to [0,1]$, rational non-negative influence function over each pair of individuals,
- $\alpha : I \to \mathbb{Z}^+$, integer non-negative effort function over the individuals,
- $\beta : I \to \mathbb{Z}^+$, integer non-negative reward function over the individuals,
- $K \in \mathbb{Z}^+$, integer non-negative value of the minimum reward required.

Output: A set $A \subseteq I$ of initially activated individuals that minimizes the global activation effort (i.e., minimizes $\sum_{i \in A} \alpha(i)$) and guarantees that the sum of the individuals rewards in the result of the influence propagation process from A is at least equals to K (i.e., $\sum_{i \in \rho^T(A)} \beta(i) \geq K$).

Being the generalized maximization version of the Max-TSS defined as:

Problem 2. Maximum weighted Effort–Reward Target Set Selection problem (Max-ER-TSS)

Input: A tuple $\langle I, \psi, \alpha, \beta, K\rangle$, where:

- I, finite non-empty set of individuals,
- $\psi : I \times I \to [0,1]$, rational non-negative influence function over each pair of individuals,
- $\alpha : I \to \mathbb{Z}^+$, integer non-negative effort function over the individuals,
- $\beta : I \to \mathbb{Z}^+$, integer non-negative reward function over the individuals,
- $K \in \mathbb{Z}^+$, integer non-negative value of the maximum effort allowed.

Output: A set $A \subseteq I$ of initially activated individuals that maximizes the sum of the individuals rewards in the result of the influence propagation process from A (i.e., maximizes $\sum_{i\in\rho^T(A)} \beta(i)$) and guarantees the global activation effort is at most equals to K (i.e., $\sum_{i\in A} \alpha(i) \le K$).

Min-ER-TSS and Max-ER-TSS are NP-hard since they generalize the Min-TSS and the Max-TSS, which are NP-hard versions of the TSS (Dreyer and Roberts, 2009; Narayanan and Wu, 2020). Next section presents new mathematical formulations for the Min-ER-TSS and the Max-ER-TSS.

3. Mathematical formulations

In this section, we propose binary linear programs to formulate the Min-ER-TSS and the Max-ER-TSS. Both problems inputs are the same tuple $\langle I, \psi, \alpha, \beta, K\rangle$, just changing the meaning of the last parameter (K). The outputs are also very similar: a set of initially activated individuals, where the Min-ER-TSS seeks to minimize the effort assuring at least some reward, while the Max-ER-TSS aims to maximize the reward guaranteeing not to exceed some effort. So, the problems are very alike and, in our formulations, we use the same variables, some identical constraints and analogous expressions for other constraints and objective functions.

Before modeling the problems, we prove the following fact which will allow us to define the variables of our formulations.

Fact 1. *Given a finite non-empty set I of individuals, a rational influence function $\psi : I \times I \to [0,1]$ over each pair of individuals and a set $A \subseteq I$ of initially activated individuals. The computation of the influence propagation process from A is described by the following sequence:*

$$A = \rho^0(A) \subset \rho^1(A) \subset \rho^2(A) \subset \cdots \subset \rho^T(A) \subseteq I,$$

where T is the first non-negative integer number for which $\rho^T(A) = \rho(\rho^T(A))$ with $T < |I|$.

Proof. If $A = \emptyset$, it follows from Definition 1 that no individual is influenced by A. Thus, $\rho(A) = \emptyset = A$ and $A = \rho^0(A) = \rho^T(A) = \emptyset \subset I$ with $T = 0 < |I|$.

If $A \neq \emptyset$, it follows from Definitions 2 and 3 that $\rho^t(A) \subseteq \rho^{t+1}(A)$ for any integer $t \ge 0$ and since the influence propagation process from A is a subset of I, we have that $\rho^T(A) \subseteq I$. Hence, we proved:

$$A = \rho^0(A) \subseteq \rho^1(A) \subseteq \rho^2(A) \subseteq \cdots \subseteq \rho^T(A) \subseteq I.$$

If $\rho^t(A) = \rho^{t+1}(A)$ for some $t < T$, then $\rho^t(A) = \rho^{t+1}(A) = \rho(\rho^t(A))$ and T would not be the first integer such that $\rho^T(A) = \rho(\rho^T(A))$, contradicting Definition 3. Therefore, $\rho^t(A) \neq \rho^{t+1}(A)$ for all $0 \le t < T$ and

$$A = \rho^0(A) \subset \rho^1(A) \subset \rho^2(A) \subset \cdots \subset \rho^T(A) \subseteq I.$$

Since $\rho^t(A) \neq \rho^{t+1}(A)$ for each $0 \le t < T$, follows that from $\rho^t(A)$ to $\rho^{t+1}(A)$ at least one new individual must be activated, and since $\left|\rho^0(A)\right| > 0$ and there are at most $|I|$ individuals, we conclude that $T < |I|$. □

Consider the sequence $A = \rho^0(A) \subset \rho^1(A) \subset \cdots \subset \rho^T(A) \subseteq I$ to compute the influence propagation process from A, as described before. We say that the individual i **is active at step** t iff $1 \le t \le T+1$ and $i \in \rho^{t-1}(A)$, or $t > T+1$ and $i \in \rho^T(A)$, defining the binary variables for our formulations as follows:[1]

$$x_i^t = \begin{cases} 1, & \text{if } i \text{ is active at step } t \\ 0, & \text{otherwise.} \end{cases}$$

Given an instance $\langle I, \psi, \alpha, \beta, K\rangle$, by Fact 1 we do not need more than $|I|$ steps to perform the computation of the influence propagation process from any subset of I. Then, considering that t takes integer values from 1 to $|I|$, the number of variables results $|I|^2$. Now, we give constraints to guarantee the feasibility of a solution over the influence propagation process. The first type of those constraints ensures that if an individual is active at some step, then he/she must remain active at further steps (once activated remains activated):

$$x_i^{t-1} \le x_i^t, \qquad \forall i \in I, 2 \le t \le |I|. \tag{1}$$

Notice that, for each individual i, the above constraints guarantee:

$$x_i^1 \le x_i^2 \le x_i^3 \le \cdots \le x_i^{|I|}.$$

Then, if an individual i is active at some step t ($x_i^t = 1$), i will remain activated for all steps $r \ge t$, since x_i^r will be equals to 1. Another group of constraints introduces the conditions to activate an individual. If the individual i is not initially activated ($x_i^1 = 0$), then, in order to be activated at step $t > 1$ ($x_i^t = 1$), i must be influenced by the active individuals in the step $t-1$, that is:

$$\sum_{j\in I\setminus\{i\}} \psi(j,i) x_j^{t-1} \ge 1.$$

Thus, for any initially non-activated individual i ($x_i^1 = 0$) the following inequality must hold at any step $t \ge 2$:

$$\sum_{j\in I\setminus\{i\}} \psi(j,i) x_j^{t-1} \ge x_i^t \qquad \forall i \in I, 2 \le t \le |I|$$

$$\equiv x_i^1 + \sum_{j\in I\setminus\{i\}} \psi(j,i) x_j^{t-1} \ge x_i^t \qquad \forall i \in I, 2 \le t \le |I|. \tag{2}$$

Observe that, if individual i is initially activated, then $x_i^1 = \cdots = x^{|I|} = 1$ and the last inequality holds true. Also, if individual i is influenced at step $t-1$ ($\sum_{j\in I\setminus\{i\}} \psi(j,i) x_j^{t-1} \ge 1$), then i must be active at step t ($x_i^t = 1$). The following constraints ensure that condition:

$$\sum_{j\in I\setminus\{i\}} \psi(j,i) \left(x_i^t - x_j^{t-1}\right) > -1 \qquad \forall i \in I, 2 \le t \le |I|.$$

The strict inequality of the previous set of constraints implies that the region of feasible solutions is not convex. So, in order to avoid strict inequalities, we introduce a very small positive value ϵ (e.g., $\epsilon = 10^{-6}$):

$$\sum_{j\in I\setminus\{i\}} \psi(j,i) \left(x_i^t - x_j^{t-1}\right) \ge -1 + \epsilon \qquad \forall i \in I, 2 \le t \le |I|. \tag{3}$$

The computation of the influence propagation process is fully described by the constraints above. The next expressions are related to the effort and the reward values of a solution. For the effort of a solution, the only significant values are associated with individuals initially activated, i.e., individuals that are active at step 1. So, the effort of a solution may be calculated as follows:

$$\sum_{i\in I} \alpha(i) x_i^1.$$

[1] We add 1 to the step indexes of the variables in order to make clear the further notation: indexes range from 1 to $|I|$ instead of from 0 to $|I|-1$.

On the other hand, for the reward of a solution, the only significant values are those associated with individuals active at the last step:

$$\sum_{i\in I}\beta(i)x_i^{|I|}.$$

Since the objective of the Min-ER-TSS is to minimize the effort of activating individuals while ensuring the reward is at least the given constant K, the below formulation is valid for the problem:

Model 1. *Binary linear program for instance* $\langle I,\psi,\alpha,\beta,K\rangle$ *of* Min-ER-TSS*:*

$$\min \sum_{i\in I}\alpha(i)x_i^1$$

$$s.t.: \textit{Constraints } (1),(2), \textit{and } (3)$$

$$\sum_{i\in I}\beta(i)x_i^{|I|}\geq K \tag{4}$$

$$x_i^t\in\{0,1\} \qquad \forall i\in I, 1\leq t\leq |I| \tag{5}$$

Our formulation for the Max-ER-TSS is very similar to the one for the Min-ER-TSS, the differences are on the expressions with effort and reward values. One of those expressions is the objective function where instead of minimizing the effort to activate the initial set as the Min-ER-TSS does, the Max-ER-TSS must maximize the reward given by the final set of activated individuals. The other expression where the formulations diverge is the constraint which uses the instance parameter K. While the Min-ER-TSS must guarantee a minimum reward value given by K, the Max-ER-TSS should ensure that the effort to activate the initial set is not greater than the allowed value K.

Model 2. *Binary linear program for instance* $\langle I,\psi,\alpha,\beta,K\rangle$ *of* Max-ER-TSS*:*

$$\max \sum_{i\in I}\beta(i)x_i^{|I|}$$

$$s.t.: \textit{Constraints } (1),(2),(3), \textit{and } (5).$$

$$\sum_{i\in I}\alpha(i)x_i^1\leq K \tag{6}$$

In order to solve Min-ER-TSS and Max-ER-TSS by using the above models, one strategy is to propose relaxations whose solution values are, respectively, lower and upper bounds of the optimal values. Such bounds may be used in enumerative algorithms to reduce the solutions' search space or in heuristics to help in the construction of solutions. In that direction, a practical and useful approach is the Lagrangian relaxation.

4. Lagrangian relaxations

In this section, we propose Lagrangian relaxations for the Min-ER-TSS and the Max-ER-TSS, providing dynamic programming algorithms to solve them. We consider that, in both problems formulations, the constraints which make hard to find an optimal solution are those ensuring the individuals active status at any step iff they were activated at the initial step or they were influenced by the set of active individuals in the previous step (constraints (2) and (3)).

Let λ and μ be non-negative real vectors associated with constraints (2) and (3), respectively. To relax the constraints (2), for each $i\in I$ and $t\in\{2,\dots,|I|\}$, we multiply $\left(x_i^t-x_i^1-\sum_{j\in I\setminus\{i\}}\psi(j,i)x_j^{t-1}\right)$ by λ_i^t, adding the resulting expression to the objective function. Analogously, constraints (3) are relaxed by adding $\mu_i^t\times\left(-1+\epsilon-\sum_{j\in I\setminus\{i\}}\psi(j,i)\left(x_i^t-x_j^{t-1}\right)\right)$ to the objective function. In this context, $\lambda\geq 0$ and $\mu\geq 0$ are called Lagrange multiplier vectors and allow us to define the Lagrangian relaxations given by Models 3 and 4.

Model 3. *Lagrangian Relaxation for Model 1 of* Min-ER-TSS *with respect to instance* $\langle I,\psi,\alpha,\beta,K\rangle$ *and multipliers* λ *and* μ*:*

$$\min \sum_{i\in I}\alpha(i)x_i^1+\sum_{t=2}^{|I|}\sum_{i\in I}\lambda_i^t\left(x_i^t-x_i^1-\sum_{j\in I\setminus\{i\}}\psi(j,i)x_j^{t-1}\right)+\sum_{t=2}^{|I|}\sum_{i\in I}\mu_i^t\left(-1+\epsilon-\sum_{j\in I\setminus\{i\}}\psi(j,i)\left(x_i^t-x_j^{t-1}\right)\right)$$

$$s.t.: \textit{Constraints } (1),(4), \textit{and } (5).$$

Model 4. *Lagrangian Relaxation for Model 2 of* Max-ER-TSS *with respect to instance* $\langle I,\psi,\alpha,\beta,K\rangle$ *and multipliers* λ *and* μ*:*

$$\max \sum_{i\in I}\beta(i)x_i^{|I|}+\sum_{t=2}^{|I|}\sum_{i\in I}\lambda_i^t\left(-x_i^t+x_i^1+\sum_{j\in I\setminus\{i\}}\psi(j,i)x_j^{t-1}\right)+\sum_{t=2}^{|I|}\sum_{i\in I}\mu_i^t\left(1-\epsilon+\sum_{j\in I\setminus\{i\}}\psi(j,i)\left(x_i^t-x_j^{t-1}\right)\right)$$

$$s.t.: \textit{Constraints } (1),(5), \textit{and } (6).$$

Since we aim to use the above Lagrangian relaxations to obtain bounds and to construct feasible solutions, an important aspect to be considered is the amount of time required to solve those relaxations. Therefore, the next subsection describes fast algorithms to solve them.

4.1. Pseudo-polynomial time dynamic programming solutions

To solve our Lagrangian relaxation Models 3 and 4, we propose dynamic programming algorithms whose main idea is to compute an optimal solution by solving a subproblem for each individual. To obtain such algorithms, we first prove in Proposition 1 the existence of an optimal solution for the Lagrangian relaxations that can be expressed in terms of optimal solutions for the individuals subproblems. Then, in Proposition 2, we show how to begin the construction of the optimal solutions for Models 3 and 4, and in Lemma 1 we give the complete algorithm. After that, the time complexity is discussed and given as a theorem.

In order to simplify the notation, from now on suppose it is given an instance $\langle I,\psi,\alpha,\beta,K\rangle$ of Min-ER-TSS (Max-ER-TSS) with fixed Lagrange multiplier vectors λ and μ. Also, assume, for each individual $i\in I$ and step t where $1\leq t\leq |I|$, the value c_i^t (d_i^t) denotes the objective function constant multiplying x_i^t of Model 3 (4). So, we may write the objective function as:

$$\min \sum_{i\in I}\sum_{t=1}^{|I|}c_i^tx_i^t=\sum_{i\in I}C_iX_i=CX \quad \left(\max \sum_{i\in I}\sum_{t=1}^{|I|}d_i^tx_i^t=\sum_{i\in I}D_iX_i=DX\right),$$

where, for each $i\in I$, $C_i=\{c_i^t\}_{t=1}^{|I|}$ ($D_i=\{d_i^t\}_{t=1}^{|I|}$) and $X_i=\{x_i^t\}_{t=1}^{|I|}$, being $C=\{C_i\}_{i\in I}$ ($D=\{D_i\}_{i\in I}$) and $X=\{X_i\}_{i\in I}$.

We define as **first optimal sub-solution** of individual $i\in I$ in Model 3 (4), any optimal solution of the following model:

Model 5. *Sub-model of Model 3 (4) for individual* $i\in I$*:*

$$\min \sum_{t=1}^{|I|}c_i^tx_i^t=C_iX_i \qquad \left(\max \sum_{t=1}^{|I|}d_i^tx_i^t=D_iX_i\right)$$

$$s.t.: \textit{Constraints } (1) \textit{ and } (5).$$

Since every x_i^t is 0 or 1 and all inequality constraints of Model 5 can be written as $x_i^1\leq x_i^2\leq\cdots\leq x_i^t$, an optimal solution can be obtained by finding the step t^* that minimizes $\sum_{t=t^*}^{|I|}c_i^t$ (maximizes $\sum_{t=t^*}^{|I|}d_i^t$). Such t^* can be found in $\mathcal{O}(|I|)$, by beginning with a solution that activates the individual at step 1 and iterates over each step t analyzing if a better solution can be achieved by not activating the individual until the step $t+1$. Algorithm 1 describes this idea.

Given a first optimal sub-solution X_i^* of individual $i\in I$, a **second optimal sub-solution** of i is an optimal solution X_i^{**} for Model 5 with

```
Input  : c_i^t: multipliers of the variables in the objective function, I: set of individuals, i:
         individual to compute sub-solutions.
Output : X_i^*, X_i^**, first and second optimal sub-solutions of i.
1  begin
2      c ← Σ_{t=1}^{|I|} c_i^t, c^* ← c, c^** ← +∞, t^* ← 1, t^** ← 0
3      for t = 1 to |I| do
4          c ← c − c_i^t
5          if c < c^** then
6              t^** ← t + 1, c^** ← c
7          end
8          if c < c^* then
9              t^** ← t^*, c^** ← c^*, t^* ← t + 1, c^* ← c
10         end
11     end
12     X_i^* ← vector with zeros from position 1 to t^* − 1 and ones from position t^* to |I|
13     X_i^** ← vector with zeros from position 1 to t^** − 1 and ones from position t^** to |I|
14     return X_i^*, X_i^**
15 end
```

Algorithm 1: Algorithm to compute first and second optimal sub-solutions of individual i. The algorithm considers the minimization case. For the maximization case, substitute the input c_i^t by d_i^t, the $+\infty$ attribution to c^{**} in Line 2 by $-\infty$ and the signs of the comparisons in lines 5 and 8.

the extra constraint that $X_i^{**} \neq X_i^*$. This sub-solution is also computed in $\mathcal{O}(|I|)$ by Algorithm 1.

The following proposition shows that there exists an optimal solution of Model 3 (4) conformed by first or second optimal sub-solutions of the individuals computed by Algorithm 1.

Proposition 1. *There exists an optimal solution $X = \{X_i\}_{i\in I}$ of Model 3 (4) such that, for each $i \in I$, if X_i^* and X_i^{**} are the outputs of Algorithm 1, then $X_i = X_i^*$ or $X_i = X_i^{**}$.*

Proof. Given an optimal solution X for Model 3 (4) if $X_i = X_i^*$ or $X_i = X_i^{**}$ for every individual $i \in I$ then the proposition is trivially true. Otherwise, let $I' \subseteq I$ be the subset containing every individual i for whom $X_i \neq X_i^*$ and $X_i \neq X_i^{**}$, where X_i^* and X_i^{**} are the outputs of Algorithm 1.

For every $i \in I$, if $x_i^{|I|} = 0$ ($x_i^1 = 1$) for one of X_i^* or X_i^{**}, then such sub-solution is composed only by zeros (ones) and, since $X_i^* \neq X_i^{**}$, it follows that $x_i^{|I|} = 1$ ($x_i^1 = 0$) for the other sub-solution. Thus, for each $i \in I'$, at least one of X_i^* and X_i^{**} satisfies that $x_i^{|I|} = 1$ ($x_i^1 = 0$). Denote by X_i' a sub-solution in $\{X_i^*, X_i^{**}\}$ that satisfies $x_i^{|I|} = 1$ ($x_i^1 = 0$) and, for each $i \in I'$, replace the values of X_i by X_i' in the solution X, obtaining a new solution X', i.e., $X' = (X \setminus \{X_i\}_{i\in I'}) \cup \{X_i'\}_{i\in I'}$. Hence, for every individual $i \in I$, X' satisfies that $X_i = X_i^*$ or $X_i = X_i^{**}$.

Since, X_i^* and X_i^{**} are solutions of Model 5, they satisfy the constraints $x_i^{t-1} \leq x_i^t$ for each $i \in I'$ and $2 \leq t \leq |I|$. Thus, X_i' also satisfies such constraints implying that X' satisfies $x_i^{t-1} \leq x_i^t$ for all individuals in I' and every step. For the individuals of $I \setminus I'$, the constraints $x_i^{t-1} \leq x_i^t$ are also satisfied by X', since they were already satisfied by X and no variable associated with them changed from X to X'.

Also, for each $i \in I'$, the selection of X_i' guarantees that $x_i^{|I|} = 1$ ($x_i^1 = 0$) in X'. So, for each $i \in I'$, the value of $x_i^{|I|}$ (x_i^1) in X' is greater (lesser) than or equal to the previous value in X and, for $i \in I \setminus I'$, the value of $x_i^{|I|}$ (x_i^1) is the same in both solutions X' and X. Therefore, the value of the expression $\sum_{j\in I} \beta(j) x_j^{|I|}$ ($\sum_{j\in I} \alpha(j) x_j^1$) for X' is greater (lesser) than or equal to its value for X, implying that X' satisfies $\sum_{j\in I} \beta(j) x_j^{|I|} \geq K$ $\left(\sum_{j\in I} \alpha(j) x_j^1 \leq K\right)$. Moreover, since X' satisfies all constraints of the Lagrangian relaxation, X' is a feasible solution for Model 3 (4).

The computation of X_i^* and X_i^{**} guarantees that $C_i X_i^* \leq C_i X_i^{**} \leq C_i X_i$ $\left(D_i X_i^* \geq D_i X_i^{**} \geq D_i X_i\right)$. Consequently, $C_i X_i' \leq C_i X_i$ $(D_i X_i' \geq D_i X_i)$, implying that $CX' \leq CX$ $(DX' \geq DX)$ and, since X is an optimal solution for Model 3 (4), X' must be an optimal solution for the same model. □

The following proposition gives us the first step in the construction of an optimal solution for the problem as described by Proposition 1.

Proposition 2. *Given the set $\{\langle X_i^*, X_i^{**}\rangle\}_{i\in I}$ of first and second optimal sub-solutions calculated by Algorithm 1 for each individual $i \in I$, there exists a solution $X = \{X_i\}_{i\in I}$ that minimizes $\sum_{i\in I} C_i X_i$ (maximizes $\sum_{i\in I} D_i X_i$) and satisfies:*

1. *$X_i \in \{X_i^*, X_i^{**}\}$ for each $i \in I$, and*
2. *$\sum_{i\in I} \beta(i) x_i^{|I|} \geq K$ $\left(\sum_{i\in I} \alpha(i) x_i^1 \leq K\right)$, and*
3. *for every $i \in I$, if X_i^* satisfies $x_i^{|I|} = 1$ ($x_i^1 = 0$), then $X_i = X_i^*$.*

Proof. Proposition 1 guarantees the existence of an optimal solution X satisfying the first two items. If X also satisfies the last item, then trivially the proposition is proved. Otherwise, there exists an individual $i \in I$, such that $x_i^{|I|} = 1$ ($x_i^1 = 0$) for X_i^* and $X_i = X_i^{**}$. By replacing in X the values of X_i^{**} by the values of X_i^*, the objective value will not be worse since $C_i X_i^* \leq C_i X_i^{**}$ ($D_i X_i^* \geq D_i X_i^{**}$) and X will still be a feasible solution, because $x_i^{|I|} = 1$ ($x_i^1 = 0$), so the value of $\sum_{i\in I} \beta(i) x_i^{|I|}$ $\left(\sum_{i\in I} \alpha(i) x_i^1\right)$ will not decrease (increase) and will continue to be greater (lesser) than or equal to K. Thus, the new solution X is an optimal solution for the problem. Therefore, we may repeat that process, at most $|I|$ times, until we get an optimal solution X satisfying the three properties of the proposition. □

By Proposition 2, after computing $\{\langle X_i^*, X_i^{**}\rangle\}_{i\in I}$, we select to X all X_i^* for which $x_i^{|I|} = 1$ ($x_i^1 = 0$) and we denote by $I' \subseteq I$, the set of individuals for whom X_i^* has $x_i^{|I|} = 0$ ($x_i^1 = 1$). To finalize the construction of an optimal solution X, we can solve the subproblem over I' and, considering the minimization version, since $x_i^{|I|} = 1$ for each $i \in I \setminus I'$, the constraint $\sum_{i\in I} \beta(i) x_i^{|I|} \geq K$ is equivalent to

$$\sum_{i\in I} \beta(i) + \sum_{i\in I'} \beta(i)\left(x_i^{|I|} - 1\right) \geq K \equiv \sum_{i\in I'} \beta(i)\left(1 - x_i^{|I|}\right) \leq \sum_{i\in I} \beta(i) - K.$$

Thus, the task is reduced to find a set $\{X_i | i \in I'\}$ such that $\sum_{i\in I'} C_i X_i$ is minimized ($\sum_{i\in I'} D_i X_i$ is maximized) and

1. $X_i \in \{X_i^*, X_i^{**}\}$ for each $i \in I'$, and
2. $\sum_{i\in I'} \beta(i)\left(1 - x_i^{|I|}\right) \leq \sum_{i\in I} \beta(i) - K$ $\left(\sum_{i\in I'} \alpha(i) x_i^1 \leq K\right)$.

In order to solve the problem above, we unequivocally identify each individual of I' by an integer in $[1, |I'|]$, and we define the dynamic programming table P with $|I'|$ rows. Each row $1 \leq j \leq |I'|$ can be indexed by any integer ℓ in $[0, \sum_{i\in I} \beta(i) - K]$ ($[0, K]$), where

- If $\ell = \sum_{i\in I', i\leq j} \beta(i)\left(1 - x_i^{|I|}\right)$ $\left(\ell = \sum_{i\in I', i\leq j} \alpha(i) x_i^1\right)$ for some sets $\{X_i | i \leq j\}$ where $i \in I'$ and each X_i is one of X_i^* or X_i^{**}, then $P[j, \ell]$ contains one of those sets that also minimizes $\sum_{i\in I', i\leq j} C_i X_i$ (maximizes $\sum_{i\in I', i\leq j} D_i X_i$).
- Otherwise, $P[j, \ell]$ contains the empty set.

From the definition of P, a set with better objective value in row $|I'|$ will give an optimal solution to the problem. Below, we describe how to populate the table P.

For row $j = 1$, at any position ℓ, $P[1, \ell]$ may contain the empty set or a single element set: $\{X_1^*\}$ or $\{X_1^{**}\}$. More precisely, since $x_1^{|I|} = 1$ ($x_1^1 = 0$) in X_1^{**}, the value of $P[1, 0]$ is equal to $\{X_1^{**}\}$ and, since $x_1^{|I|} = 0$ ($x_1^1 = 1$) in X_1^*, the value of $P[1, \beta(1)]$ is equal to $\{X_1^*\}$ ($P[1, \alpha(1)] = \{X_1^*\}$), being $P[1, \ell]$ the empty set for every other position ℓ:

$$P[1, \ell] = \begin{cases} X_1^{**}, & \text{if } \ell = 0 \\ X_1^*, & \text{if } \ell = \beta(1)\ (\ell = \alpha(1)) \\ \emptyset, & \text{otherwise.} \end{cases}$$

At any further row $1 < j \leq |I'|$, the value of $P[j, \ell]$ depends on the values of the previous row $j - 1$. If $\ell < \beta(j)$ ($\ell < \alpha(j)$), then $x_j^{|I|}$ (x_j^1) must be 1 (0), since it would be impossible to satisfy $\ell = \sum_{i\in I', i\leq j} \beta(i)\left(1 - x_i^{|I|}\right)$ $\left(\ell = \sum_{i\in I', i\leq j} \alpha(i) x_i^1\right)$. Hence, for $\ell < \beta(j)$ ($\ell < \alpha(j)$) where $P[j-1, \ell] \neq \emptyset$:

$$P[j, \ell] = P[j-1, \ell] \cup \{X_j^{**}\}.$$

If $P[j-1,\ell] = \emptyset$, then $P[j,\ell] = \emptyset$ since there are no solutions satisfying $\ell = \sum_{i\in I',i\leq j} \beta(i)\left(1-x_i^{|I|}\right)$ $\left(\ell = \sum_{i\in I',i\leq j} \alpha(i)x_i^1\right)$.

For $\ell \geq \beta(j)$ $(\ell \geq \alpha(j))$, suppose $X = \{X_i | i \leq j\}$ is a solution that satisfies $\ell = \sum_{i\in I',i\leq j} \beta(i)\left(1-x_i^{|I|}\right)$ $\left(\ell = \sum_{i\in I',i\leq j} \alpha(i)x_i^1\right)$ and minimizes $\sum_{i\in I',i\leq j} C_iX_i$ (maximizes $\sum_{i\in I',i\leq j} D_iX_i$). Then, there are two possible scenarios:

- $X_j^* \in X$, implying that $X \setminus \left\{X_j^*\right\}$ is an optimal solution that satisfies $\ell - \beta(j) = \sum_{i\in I',i<j} \beta(i)\left(1-x_i^{|I|}\right)$ $\left(\ell - \alpha(j) = \sum_{i\in I',i<j} \alpha(i)x_i^1\right)$, i.e., $P[j-1,\ell-\beta(j)] = X \setminus \left\{X_j^*\right\}$ $\left(P[j-1,\ell-\alpha(j)] = X \setminus \left\{X_j^*\right\}\right)$.
- Otherwise, $X_j^{**} \in X$, implying that $X \setminus \left\{X_j^{**}\right\}$ is an optimal solution that satisfies $\ell = \sum_{i\in I',i<j} \beta(i)\left(1-x_i^{|I|}\right)$ $\left(\ell = \sum_{i\in I',i<j} \alpha(i)x_i^1\right)$, i.e., $P[j-1,\ell] = X \setminus \left\{X_j^{**}\right\}$.

Therefore, $P[j,\ell]$ is the best solution between $P[j-1,\ell] \cup \left\{X_j^{**}\right\}$ and $P[j-1,\ell-\beta(j)] \cup \left\{X_j^*\right\}$ $\left(P[j-1,\ell-\alpha(j)] \cup \left\{X_j^*\right\}\right)$. Moreover, if $P[j-1,\ell] = \emptyset$ and $P[j-1,\ell-\beta(j)] = \emptyset$ $(P[j-1,\ell-\alpha(j)] = \emptyset)$, then $P[j,\ell] = \emptyset$ because the best solutions cannot be constructed by a previous empty set.

From the above discussion, we conclude the following lemma to compute the dynamic table P.

Lemma 1. *Considering $\gamma = \beta$ $(\gamma = \alpha)$, the dynamic table P can be computed at each row $1 \leq j \leq |I'|$ and column $0 \leq \ell \leq \sum_{i\in I} \beta(i) - K$ $(0 \leq \ell \leq K)$ as follows:*

- *If $j = 1$, then* $P[j,\ell] = \begin{cases} X_1^{**}, & \text{if } \ell = 0 \\ X_1^*, & \text{if } \ell = \gamma(1) \\ \emptyset, & \text{otherwise.} \end{cases}$
- *Else, if $\ell < \gamma(j)$, then* $P[j,\ell] = \begin{cases} P[j-1,\ell] \cup X_1^{**}, & \text{if } P[j-1,\ell] \neq \emptyset \\ \emptyset, & \text{otherwise.} \end{cases}$
- *Else, if $P[j-1,\ell] \neq \emptyset$ and $P[j-1,\ell-\gamma(j)] = \emptyset$ or the objective function value of $P[j-1,\ell] \cup \left\{X_j^{**}\right\}$ is better than the objective function value of $P[j-1,\ell-\gamma(j)] \cup \left\{X_j^*\right\}$, then $P[j,\ell] = P[j-1,\ell] \cup X_1^{**}$.*
- *Else, if $P[j-1,\ell-\gamma(j)] \neq \emptyset$, then $P[j,\ell] = P[j-1,\ell-\gamma(j)] \cup X_1^*$.*
- *Else $P[j,\ell] = \emptyset$.*

To solve Model 3 (4), we can compute the optimal sub-solutions for each individual in time $\mathcal{O}(|I|)$, resulting in $\mathcal{O}(|I|^2)$ overall time complexity. The time required to select the sub-solutions that satisfy the last property of Proposition 2 is $\mathcal{O}(|I|)$. Then, an algorithm based on Lemma 1 can compute each row of the dynamic table with $\mathcal{O}(\sum_{i\in I} \beta(i) - K)$ $(\mathcal{O}(K))$ elementary operations and since there are $|I'| \leq |I|$ rows, the computation of table T will take $\mathcal{O}\left(|I| \times \left(\sum_{i\in I} \beta(i) - K\right)\right)$ $(\mathcal{O}(|I| \times K))$ time. Finally, we select the best solution in the last row of the table in time $\mathcal{O}\left(\sum_{i\in I} \beta(i) - K\right)$ $(\mathcal{O}(K))$ which is an optimal solution for the problem. This discussion leads us to the following theorem, which states the existence of a pseudo-polynomial time algorithm for Model 3 (4).

Theorem 1. *Model 3 (4) can be exactly solved in $\mathcal{O}\left(|I|^2 + |I| \times M\right)$ time, where $M = \sum_{i\in I} \beta(i) - K$ $(M = K)$.*

4.2. Quality of the relaxations

Besides the computational efficiency, another important property of relaxations is related to the quality of the bounds provided by its solutions. In that direction, we show that there exist multipliers λ and μ for whom the bounds of our Lagrangian relaxation are strictly better than the bounds of a linear relaxation of Models 1 and 2.

Fisher (1981) gave sufficient conditions for a Lagrangian relaxation be better than a linear relaxation. Those conditions imply that a Lagrangian relaxation offers strictly better bounds than a linear relaxation if there are no integer optimal solutions for the Lagrangian relaxation when the integrality constraints are dropped. Thus, to prove the existence of Lagrange multipliers λ and μ, for whom the bounds given by Model 3 (4) are strictly greater (lesser) than the bounds obtained by a linear relaxation of Model 1 (2), we show that, for those λ and μ, no optimal solution of the linear relaxation of Model 3 (4) satisfies the integrality constraints.

If the linear relaxation of Model 3 (4) has an optimal solution that satisfies the integrality constraints, then that solution objective value is equals to the value of solution X calculated by the dynamic programming algorithm described in the previous subsection. Thus, X is an optimal solution of the linear relaxation and satisfies Proposition 1. Hence, given the set of first and second optimal sub-solutions $\left\{\left\langle X_i^*, X_i^{**}\right\rangle\right\}_{i\in I}$ calculated by Algorithm 1, we formulate the problem described in Proposition 1, whose solution is X. In that formulation, we consider, for each individual $i \in I$, the binary variable y_i, that takes value 1 if X_i^* is part of the solution X and value 0 if X_i^{**} is part of the solution X:

Model 6.

$$\min \sum_{i\in I} C_i\left(X_i^* y_i + X_i^{**}(1-y_i)\right) \quad \left(\max \sum_{i\in I} D_i\left(X_i^* y_i + X_i^{**}(1-y_i)\right)\right)$$

$$s.t.:$$

$$\sum_{i\in I} \beta(i)\left(x_i^{*|I|} y_i + x_i^{**|I|}(1-y_i)\right) \geq K$$

$$\left(\sum_{i\in I} \alpha(i)\left(x_i^{*1} y_i + x_i^{**1}(1-y_i)\right) \leq K\right)$$

$$y_i \in \{0,1\} \quad \forall i \in I.$$

Any solution $Y = \left\{y_i\right\}_{i\in I}$ of the linear relaxation of the above model will produce a feasible solution $XY = \left\{X_i^* y_i + X_i^{**}(1-y_i) | i \in I\right\}$ of the linear relaxation of Model 3 (4). Constraints of type $x_i^t \leq x_i^{t+1}$ are satisfied because $x_i^{*t} \leq x_i^{*t+1}$ and $x_i^{**t} \leq x_i^{**t+1}$, thus $x_i^{*t} y_i + x_i^{**t}(1-y_i) \leq x_i^{*t+1} y_i + x_i^{**t+1}(1-y_i)$ for any $y_i \in [0,1]$. The constraint $\sum_{i\in I} \beta(i) x_i^{|I|} \geq K$ $\left(\sum_{i\in I} \alpha(i) x_i^1 \leq K\right)$ was already satisfied by Model 6. Furthermore, the objective value associated with Y in Model 6 is equal to the objective value associated with XY in Model 3 (4). Therefore, the value of an optimal solution of the linear relaxation of Model 3 (4) is lesser (greater) than or equal to the value of an optimal solution of the linear relaxation of Model 6.

If we define $c_i = C_i(X_i^* - X_i^{**})$ $(d_i = D_i(X_i^* - X_i^{**}))$, $b_i = \beta(i)(x_i^{*1} - x_i^{**1})$ $(a_i = \alpha(i)(x_i^{*1} - x_i^{**1}))$ and $KB = K - \sum_{i\in I} \beta(i) x_i^{**}$ $(KA = K - \sum_{i\in I} \alpha(i) x_i^{**})$ for each individual $i \in I$, then Model 6 may be rewritten as:

$$\min \sum_{i\in I} c_i y_i \qquad \left(\max \sum_{i\in I} d_i y_i\right)$$

$$s.t.:$$

$$\sum_{i\in I} b_i y_i \geq KB \qquad \left(\sum_{i\in I} a_i y_i \leq KA\right)$$

$$y_i \in \{0,1\} \quad \forall i \in I.$$

The above problem is the binary Knapsack problem (Martin, 1999) and, in general, there are no optimal solutions of the linear relaxation where all variables take integer values. Thus, for general cases, the value of an optimal solution Y of the linear relaxation is strictly lesser (greater) than the value of X. Hence, the value of XY is also lesser (greater) than or equal to the value of X and X cannot be an optimal solution of the linear relaxation of Model 3 (4). Therefore, for general

instances and all non-negative real values of Lagrange multipliers λ and μ, there are no optimal solutions of the linear relaxation of Model 3 (4) that satisfy the integrality constraints. Moreover, for general instances, our Lagrangian relaxation is guaranteed to provide a value strictly greater (lesser) than the optimum value of the linear relaxation of Model 1 (2). So, we have the following theorem.

Theorem 2. *For general* Min-ER-TSS *(*Max-ER-TSS*) instances, there exist non-negative real Lagrange multipliers λ and μ for which the optimal solution value of Model 3 (4) is strictly greater (lesser) than the optimal solution value of the linear relaxation of Model 1 (2).*

In the next section, we describe a subgradient method to find the values of λ and μ, that guarantee better bounds than the linear relaxation.

5. Subgradient method

Subgradient methods have been largely used to find Lagrange multipliers for which the bounds are as good as possible. Zhao et al. (1999) describe general subgradient methods, proving their correctness and efficiency. We adapted the subgradient method to not only obtain the bounds but also to compute a feasible heuristic solution for the problem. Algorithm 2 shows the subgradient method we use, being the Lagrangian coefficient matrix $Q \in \mathbb{R}^{2|I|^2 \times |I|^2}$ and independent terms vector $r \in \mathbb{R}^{2|I|^2}$ defined over the relaxed constraints:

$$x_i^1 + \sum_{j \in I \setminus \{i\}} \psi(j,i) x_j^{t-1} - x_i^t \geq 0 \ \ \forall i \in I, 2 \leq t \leq |I| \tag{7}$$

$$\sum_{j \in I \setminus \{i\}} \psi(j,i) \left(x_i^t - x_j^{t-1} \right) \geq -1 + \epsilon \ \ \forall i \in I, 2 \leq t \leq |I|, \tag{8}$$

where, the first $|I|^2$ lines of Q correspond to the coefficients of the variables of inequalities (7) and the last $|I|^2$ lines to the coefficients of the variables of inequalities (8). Similarly, the first $|I|^2$ values of r correspond to the independent terms of inequalities (7), while the last $|I|^2$ elements are the independent terms of inequalities (8).

Algorithm 2 describes a standard subgradient optimization method (see Zhao et al., 1999) with the inclusion of initially generated combinatorial bounds and greedy feasible solutions (lines 6 and 7), and a heuristic procedure to obtain feasible solutions (line 25) from the Lagrangian relaxation ones.

For the initial combinatorial lower (upper) bound we calculate two values: m, the minimum number of active individuals required to activate at least one new individual and m_k, the minimum (maximum) number of individuals whose reward (effort) sum is greater (lesser) than or equal to K. If $m_k < m$, then any feasible solution of Min-ER-TSS (Max-ER-TSS) must (can) activate at least (most) m_k individuals in order to satisfy an activation reward (effort) greater (lesser) than or equal to K. In such case, an initial combinatorial lower (upper) bound is the effort (reward) sum of the m_k individuals with lesser (greater) effort (reward) values. If $m_k \geq m$, then by activating m individuals, the activation process may begin and in some cases could reach the complete network. Thus, when $m_k \geq m$, an initial combinatorial lower (upper) bound is the effort sum of the m individuals with lesser effort values (the reward sum of the complete network).

The initial feasible solution of Algorithm 2 is greedily constructed. For the Min-ER-TSS, the construction begins with an empty set of initially activated individuals. Then, until the reward of the activated individuals is lesser than K, we select to the initially activated set the non-activated individual i with greater $\frac{\beta(i)}{\alpha(i)}$ value and apply the activation process. For the Max-ER-TSS, the idea is similar, but the condition to stop is when no new individual can be added to the set

```
   Input  : ⟨I, ψ, α, β, K⟩, instance of Min-ER-TSS and stop conditions' parameters ξ1, ξ2 and ξ3.
   Output : Xl, lb, Xu, ub, solution and value of the Lagrangian relaxation with best lower bound
            found and feasible heuristic solution with its associated value.
1  begin
2      Q ← Lagrangian coefficient matrix for I, ψ
3      r ← Lagrangian independent terms vector for I, ψ
4      λ, μ ← Initial Lagrange multipliers vector
5      s ← Initial step size
6      Xl, lb ← Initial combinatorial lower bound and solution associated
7      Xu, ub ← Initial feasible combinatorial solution and its value
8      g ← +∞
9      Start counter for iterations without change
10     while s > ξ1 and ||g||^2 > ξ2 and ub − lb > ξ3 do
11         X', v' ← Optimal solution and objective value of Model 3 with Lagrange multipliers λ
               and μ
12         if v' > lb then
13             Xl, lb ← X', v'
14         else
15             Increment counter for iterations without change
16             if counter for iterations without change reached a limit then
17                 s ← θ × s
18                 Restart counter for iterations without change and redefine its limit
19             end
20         end
21         g ← Q × X' − r
22         if ||g||^2 > 0 then
23             λ, μ ← (λ, μ) + (s×(ub−v')) / ||g||^2 × g
24         end
25         X', v' ← Compute new feasible combinatorial solution from X', v'
26         if v' < ub then
27             Xu, ub ← X', v'
28         end
29     end
30     return Xl, lb, Xu, ub
31 end
```

Algorithm 2: Subgradient method for Model 3. ξ_1, ξ_2 and ξ_3 are the stop conditions parameters of the algorithm, and they should be *"small enough"* to guarantee a *"good enough"* precision for the solutions. The parameter θ is used to update the step size multiplier (usually to decrease the value when the steps are too large). For the maximization case, instead of computing a combinatorial lower bound at Line 6, compute a feasible combinatorial solution, while in Line 7 compute a combinatorial upper bound ub and in Line 11 solve the Lagrangian relaxation given by Model 4. To finalize the algorithm adaptation for the maximization case, swap the comparisons of lines 12 and 26 and the updates of lines 13 and 27.

of initially activated individuals without getting an effort sum greater than K.

At each iteration of the subgradient method, after computing the Lagrangian bounds by using the pseudo-polynomial time dynamic programming algorithm discussed in previous sections (line 11), a new heuristic solution is constructed (line 25) following the same idea of the greedy algorithm used to generate the initial feasible solution. The difference is that initially activated individuals of the last solution of the Lagrangian relaxation have priority to be included in the initially activated set of the new heuristic solution.

In the next section, we present tests of the described subgradient method over instances constructed from real-world scenarios.

6. Computational experiments

To test our proposals, we generated instances based on real-world datasets provided by Batagelj and Mrvar (2006). We solved each instance with our subgradient method and compared the bounds obtained with the bounds given by the linear relaxation solutions of Models 1 and 2. For each execution of the subgradient method we fixed the parameters as follows: $\xi_1 = \xi_2 = 10^{-4}$, $\xi_3 = 9 \times 10^{-1}$, $\theta = 0.5$, initial $\lambda = \mu =$ null vector, $s = 2$, initial limit of iterations without improving the bound equals to $\min\{30, |I|\}$, and the limit of iterations without improving the bound redefinition was set equal to $\frac{2}{3}$ of its previous value.

We also implemented a Branch & Bound algorithm where, at each explored node, the algorithm calls our subgradient method to calculate bounds and also to construct a heuristic solution trying to improve the best solution found. The decision of which node to explore is given

Table 1
Selected datasets from Batagelj and Mrvar (2006). Columns `Symmetrical`, $|I|$ and `Relations` indicate, respectively, if the relations are symmetrical, the number of individuals and the number of entries in the relation matrices.

Number	Name: Datasets	Symmetrical	$\|I\|$	Relations
1	Knoke Bureaucracies: KNOKI	No	10	100
2	Knoke Bureaucracies: KNOKM	No	10	100
3	Roethlisberger & Dickson Bank: RDGAM	Yes	14	196
4	Roethlisberger & Dickson Bank: RDPOS	Yes	14	196
5	Roethlisberger & Dickson Bank: RDHLP	No	14	196
6	Kapferer Mine: KAPFMU	Yes	15	225
7	Kapferer Mine: KAPFMM	Yes	15	225
8	Thurman Office: THURA	No	15	225
9	Thurman Office: THURM	Yes	15	225
10–24	Newcomb Fraternity: NEWC1 … NEWC15	No	17	289
25	Davis Southern Club Women: DAVIS	Yes	18	252
26–28	Sampson Monastery: SAMPLK1 … SAMPLK3	No	18	324
29	Sampson Monastery: SAMPES	No	18	324
30	Sampson Monastery: SAMPIN	No	18	324
31–51	Krackhardt Office css: KRACKAD1 … KRACKAD21	No	21	441
52–72	Krackhardt Office css: KRACKFR1 … KRACKFR21	Yes	21	441
73	Zachary Karate Club: ZACHE	Yes	34	1156
74	Zachary Karate Club: ZACHC	Yes	34	1156
75	Bernard & Killworth Technical: BKTECC	No	34	1156
76–77	Kapferer Tailor Shop: KAPFTI1, KAPFTI2	No	39	1521
78–79	Kapferer Tailor Shop: KAPFTS1, KAPFTS2	Yes	39	1521
80	Bernard & Killworth Office: BKOFFC	No	40	1600
81	Bernard & Killworth Ham Radio: BKHAMC	No	44	1936
82	Bernard & Killworth Fraternity: BKFRAC	No	58	3364

by a variable with the greater number of constraints unsatisfied by the Lagrangian relaxation and the first explored value (zero or one) depends on the value of such variable on the best solution found until that moment by the Branch & Bound (an attempt of following the path described by the best solution found). We tested all instances with our Branch & Bound comparing at each branching the bounds given by the subgradient method and the linear relaxation.

Since Branch & Bound are enumerative algorithms and an exact solution may take an enormous amount of time to be found, we set a time limit for the executions equals to one hour. We also solved the instances with the Gurobi solver (Gurobi Optimization, 2020), after formulating them with Models 1 and 2. To compare the performance of the proposed Lagrangian relaxation with respect to linear relaxations embedded in enumerative methods, we also set one hour as the time limit for the Gurobi solver execution, disabling the solver heuristics and cuts generation.

The rest of this section describes the computational environment for the experiments, the selected datasets and the instance construction process, finishing with the tests results and their analysis.

6.1. Computing environment

The computational resources used in the experiments were: Processor Intel® Corep™ i5-7400M (CPU @ 3.0 GHz and 8 GB of RAM); Operational system Microsoft Windows 10 Pro 64 bits; Programming language Python 3.7. The CPU times were obtained by using the `time()` function from Python's `time` module.

6.2. Test instances

The datasets were obtained from *UCINET IV DATASETS* (Batagelj and Mrvar, 2006). All selected datasets are described by Table 1.

The datasets informed in Table 1 report relations between individuals of the same community observed in real-world studies. Those relations represent the number of times each pair of individuals interacted in a task of interest, the ranking that each individual gives to the others and the existence of reliability or friendship between the individuals. Hence, to construct the instances from the databases we compute, for each individual $i \in I$, a value R_i as the sum of the relation values of the other individuals over i (i.e., $R_i = \sum_{j \in I \setminus \{i\}} relation(j,i)$). Then, for each pair of individuals i, j, we define $\psi(j,i) = 2 \times \frac{relation(j,i)}{R_i}$, guaranteeing that an individual i will be activated iff the activated relation sum over i is at least $\frac{R_i}{2}$, which simulates the majority threshold activation (Ackerman et al., 2010).

In order to mimic more realistic scenarios, we consider that if an individual $i \in I$ requires a "small" number of already activated individuals to be activated, then i should require a small effort to be initially activated. That situation may be modeled by counting for each individual $i \in I$, the number of individuals with influence value over i greater than or equal to some threshold (e.g., $\frac{1}{|I|}$). Thus, for each individual $i \in I$, we define the effort as $\alpha(i) = 1 + |I| - \left|\left\{ j \mid j \in I \setminus \{i\} \text{ and } \psi(j,i) \geq \frac{1}{|I|} \right\}\right|$.

We also assume that individuals with greater influence over the others, usually provide greater rewards. Therefore, for each individual $i \in I$, we define the reward as $\beta(i) = 1 + \left\lfloor \sum_{j \in I \setminus \{i\}} \psi(i,j) \right\rfloor$.

Finally, for the Min-ER-TSS we set $K = \left\lceil \frac{\sum_{i \in I} \beta(i)}{2} \right\rceil$, to guarantee that every solution has at least half of the total reward. For Max-ER-TSS we set $K = \left\lceil 2 \times \frac{\sum_{i \in I} \alpha(i)}{|I|} \right\rceil$, such a formula was defined to avoid *"easy"* instances where optimal values are very close to those obtained by the linear relaxation. Table 2 gives the number of variables and constraints of the formulations for each dataset.

6.3. Results and analysis

To analyze the quality of the bounds, for each instance we solved the linear relaxation using the Gurobi solver and we also computed the bound given by our subgradient method approach. The results of these tests are illustrated by Fig. 1 for the Min-ER-TSS and by Fig. 2 for the Max-ER-TSS.

From Fig. 1 it is evident the superiority of the lower bounds found by our subgradient method in comparison with the bounds obtained by the linear relaxation of Model 1, while the computational times required by both approaches were very low and similar for the majority of the experiments. For Min-ER-TSS we observe that the linear relaxation

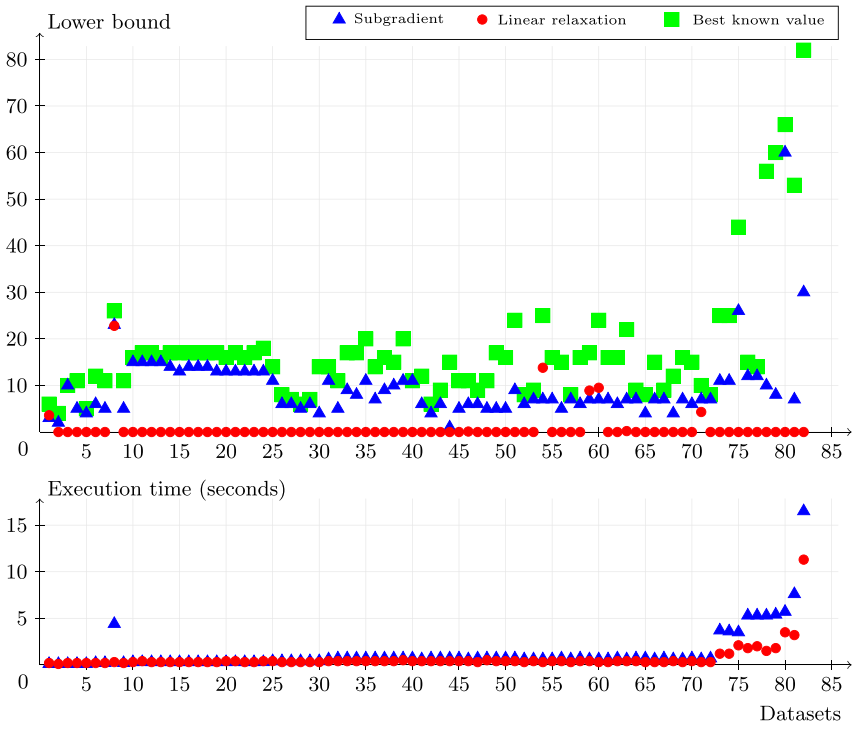

Fig. 1. Bounds and execution times of our subgradient approach and the linear relaxation of Model 1 for MIN-ER-TSS on each dataset. Figure generated from Table A.3 of Appendix, the best known values were taken from Tables A.7 and A.8.

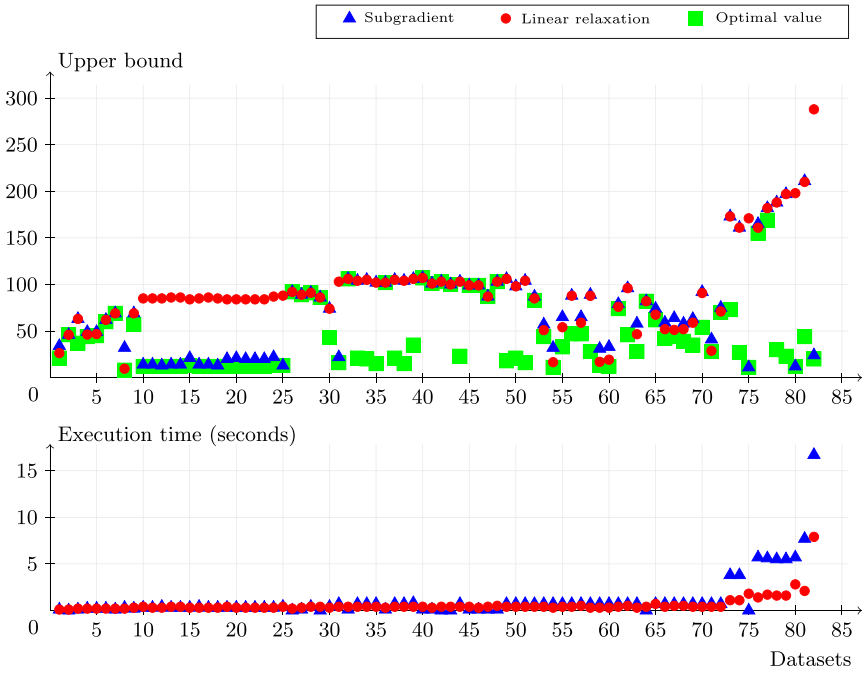

Fig. 2. Bounds and execution times of our subgradient approach and the linear relaxation of Model 2 for MAX-ER-TSS on each dataset. Figure generated from Table A.4 of Appendix, the best known values were taken from Tables A.9 and A.10.

Table 2
Number of variables and constraints of Models 1 and 2 for each dataset.

Number	Datasets	Variables	Constraints
1–2	KNOK and KNOKM	100	271
3–5	RDGAM, RDPOS and RDHLP	196	547
6–9	KAPFMU, KAPFMM, THURA and THURM	225	631
10–24	NEWC1 … NEWC15	289	817
25–30	DAVIS, SAMPLK1 … SAMPLK3, SAMPES and SAMPIN	324	919
31–72	KRACKAD1 … KRACKAD21 and KRACKFR1 … KRACKFR21	441	1261
73–75	ZACHE, ZACHC and BKTECC	1156	3367
76–79	KAPFTI1, KAPFTI2, KAPFTS1 and KAPFTS2	1521	4447
80	BKOFFC	1600	4681
81	BKHAMC	1936	5677
82	BKFRAC	3364	9919

lower bounds were worse than the subgradient bounds for 95.1% of the tested instances, where the average linear relaxation bound as percent of the subgradient bound was 2.2%. Only in 4.9% of the tested Min-ER-TSS instances, the linear relaxation obtained better bounds than those given by our subgradient method, and even in those cases the bounds of the subgradient were greater than 50% of the linear relaxation bound, being 71.6% the average subgradient bound as percent of the linear relaxation bound (for these 4.9% instances). However, since the linear relaxation of Model 1 resulted in the value of zero for 90.2% of the tests, it may imply that our formulation for the Min-ER-TSS is not strong enough and that better formulations should be considered for this version of the problem.

Analyzing the results reported by Fig. 2 for Max-ER-TSS, we note that both strategies obtained the same bounds in 45.1% of the tested instances, being the bounds of the linear relaxation better than those from the subgradient method in 30.5% of the cases, while the subgradient bounds were better than the linear relaxation bounds in 24.4% of the tests. Notice that for the 30.5% of the tests where the subgradient was worse, the average value of the subgradient bounds as percent of the linear relaxation bound was 128.7%, while for the 24.4% where the subgradient was better, the average value of the linear relaxation bound as percent of the subgradient bound was 696.1%, implying that the quality of the subgradient bounds was (in general) superior to that of the linear relaxation bounds. Such a conclusion is enforced when we analyze all tested instances, where the average subgradient bound as percent of the linear relaxation bound was 88.7%. Therefore, even when the linear relaxation obtained better bounds for more instances, the average values of the subgradient bounds were much tighter than those of the linear relaxation bounds. Analogously to the Min-ER-TSS, the computational times of both approaches were very low and similar for the majority of the experiments.

A better comprehension of the subgradient bounds quality can be achieved if compared with the optimal values for each instance. From Fig. 1 we observe that, for Min-ER-TSS, the subgradient bounds were equal to the optimal values for 2.4% of the tests, above 80% of the optimal values for 26.8% of the tests, and at least 50% of the optimal values for 62.2% of the tests. Although the subgradient bounds were better than the linear relaxation bounds for Min-ER-TSS and tight enough for most of these instances, there was a subset representing 4.8% of the tests where the subgradient bounds were very low, less than 20% of the optimal value. For Max-ER-TSS, Fig. 1 shows that the subgradient bounds were equal to the optimal values in 24.4% of the tests, below 140% of the optimal values in 52.2% of the tests, and at most 200% of the optimal values for 73.2% of the tests. Despite the subgradient was capable of obtaining more tighter bounds for Max-ER-TSS, the subset of instances with very inaccurate bounds (above five times the optimal values) was larger than the one for Min-ER-TSS, representing 9.8% of the tested instances.

For a deeper comparison of the sharpness of the bounds, we executed a Branch & Bound algorithm, where at every node the subgradient method was executed and the linear relaxation was solved. Fig. 3 illustrates, for each instance, the average values of the linear relaxation bound as a percent of our subgradient method bound on every node, and the average execution time of the linear relaxation as a percent of the subgradient execution time.

Fig. 3 shows that, for the Min-ER-TSS, the lower bounds provided by the subgradient method during the Branch & Bound execution were much sharper than the lower bounds obtained by the linear relaxation of Model 1. That observation is based on the fact that, for almost all tests (over 90%), the lower bound of the linear relaxation was under a 20% of the lower bound of the subgradient method. Besides that, the superiority of our subgradient method for the Min-ER-TSS on the tested datasets is also given by the computational efficiency, since for almost all instances the linear relaxation execution took over 200% the time required by our subgradient method. An analogous analysis can be done for Max-ER-TSS, where for more than a half of the instances (over 67%) the linear relaxation upper bound was ten times worse than the subgradient upper bound, being also superior the computational efficiency of the subgradient approach achieving (for some instances) a time lesser than 1% of the linear relaxation time.

The subgradient method is also designed to construct heuristic solutions for the problems. So, we analyze the quality of the obtained solutions for each tested dataset, executing also the Branch & Bound with our subgradient method embedded to exactly solve the instances. Figs. 4 and 5 illustrate, for each instance of Min-ER-TSS and Max-ER-TSS, respectively, our heuristic solution value and the solution value obtained by the proposed Branch & Bound algorithm and the Gurobi solver.

From Fig. 4 we observe that for 51.2% of Min-ER-TSS instances our subgradient method obtained heuristic solutions whose values were optimal, for another 34.1% of the tests the gap[2] was less than 20%, and only for 2.4% of the tested instances the gap was greater than 50%. For Max-ER-TSS the results in Fig. 5 are very similar, in 53.7% of the instances the subgradient method obtained optimal solutions, in 31.7% of the tests the gap was less than 20%, and only for 2.4% of the instances the presented gap was greater than 50%.

To evaluate the performance of a Branch & Bound based on the Lagrangian relaxation and an exact solver based on linear relaxation, we may analyze the results of the implemented Branch & Bound and the Gurobi solver in Figs. 4 and 5. For both problems, we notice that both approaches were able to solve to optimality almost all instances, but the average time required by the Gurobi solver was 193.2 s while for the implemented Branch & Bound this value was slightly greater (208.8 s). Although the average computational time of the Gurobi solver was smaller, the implemented Branch & Bound usually explored much fewer nodes of the enumeration tree, being the average number of explored nodes by the Branch & Bound equal to 523.8 while for the

[2] The gap expression is $\frac{|v-BKV|\times 100}{BKV}$, where v is the value of the subgradient and BKV the best known value

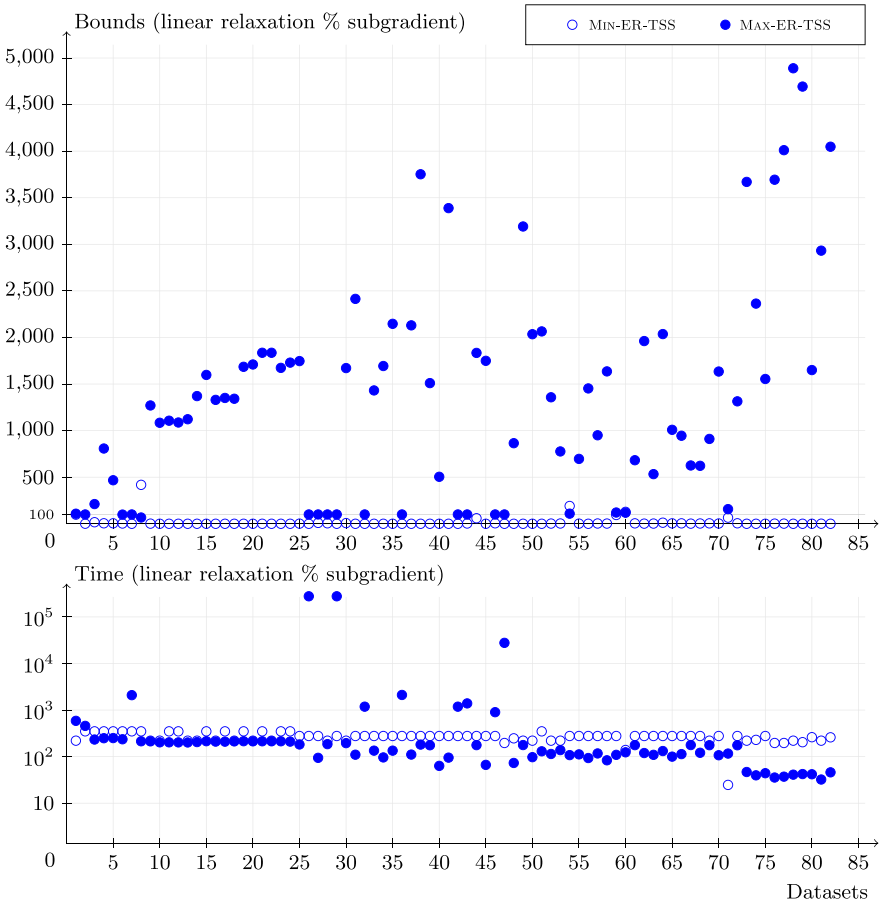

Fig. 3. Average values of the linear relaxation bounds as a percent of the bounds provided by the subgradient method during the Branch & Bound execution for each dataset and average linear execution times as a percent of the average execution times given by the subgradient method. Figure generated from Tables A.5 and A.6 of Appendix.

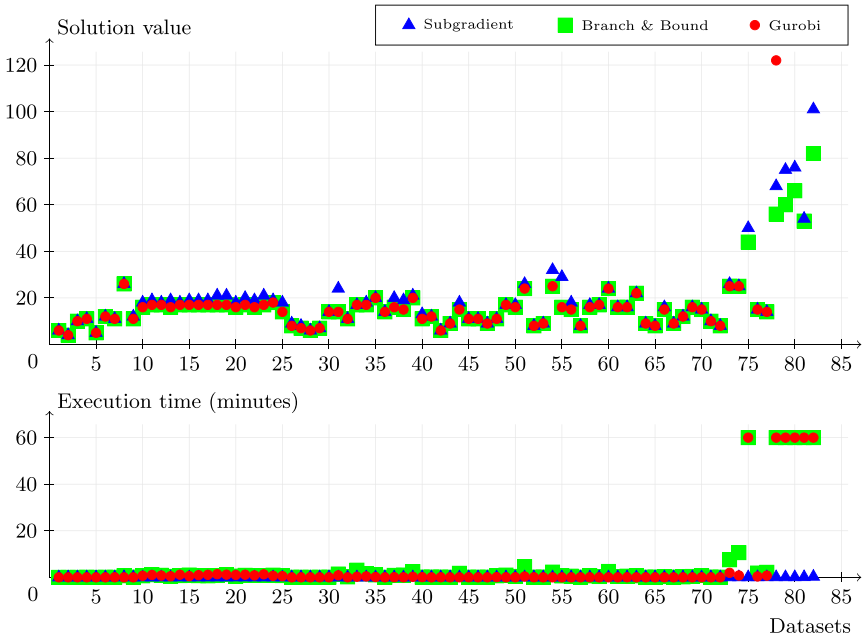

Fig. 4. Execution results of our subgradient approach, Branch & Bound and the Gurobi solver over each instance of the Min-ER-TSS. The first graphic shows the solution values found by each approach. The second graphic shows the execution time of each approach in minutes. Figure generated from Tables A.7 and A.8 of Appendix.

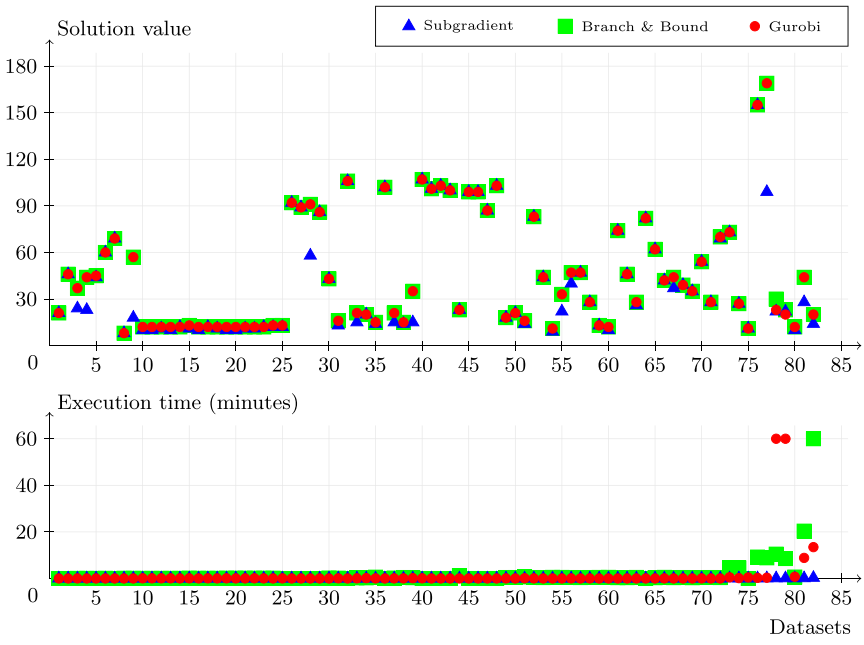

Fig. 5. Execution results of our subgradient approach, Branch & Bound and the Gurobi solver over each instance of the Max-ER-TSS. The first graphic shows the solution values found by each approach. The second graphic shows the execution time of each approach in minutes. Figure generated from Tables A.9 and A.10 of Appendix.

Gurobi solver this value was 25 876.8. Therefore, to propose robust and competitive exact algorithms that use our Lagrangian relaxation, some strategies should be introduced that consider preprocessing, to reuse computations in previous subgradient calls, to reduce the number of nodes where the subgradient method is executed, among others. Despite the implemented Branch & Bound must be improved to be considered a proposal for solving these problems, we notice that this method obtained better gaps and solutions than the Gurobi solver (without the heuristics and cut generations) in 4.9% of the experiments, while the Gurobi solver only obtained a better gap in 0.6% of the experiments (one instance: 82 − BKFRAC for Max-ER-TSS).

In general, our subgradient method is not only very fast but also provided nice bounds and heuristic solutions for many of the tested instances, being an interesting approach for Min-ER-TSS and Max-ER-TSS. Besides, this method can be easily embedded within other techniques as metaheuristics or enumerative algorithms.

7. Final comments and future directions

This paper considered the Min-ER-TSS and the Max-ER-TSS which are NP-hard problems that generalize already studied versions of the TSS and allow to model a larger diversity of real-world scenarios. For each problem, we gave a binary linear model and proposed Lagrangian relaxations to compute their bounds. We also designed efficient algorithms based on dynamic programming techniques to solve the Lagrangian relaxations and proved that, for general cases, the Lagrangian relaxation bounds are strictly better than the bounds given by a linear relaxation of our binary formulations. Besides, we designed a subgradient method to find Lagrangian bounds with a good enough precision, which had embedded a greedy heuristic to compute feasible solutions.

To test our approaches, we generated instances from real-world datasets and compared with the linear relaxation of the Gurobi solver (Gurobi Optimization, 2020). We also implemented a Branch & Bound algorithm that at each node used the proposed subgradient method to decide if the node would be explored or not.

The computational experiments show that for both problems our subgradient method obtained competitive bounds when compared with those given by the linear relaxation, and for some instances the bounds were equal to the optimal value. Also, the heuristic solutions of the subgradient method were optimal for many instances (over 50% of the tests), with optimality guarantees for some of them. Besides that, the subgradient method was very fast with execution times less than one second for a majority of the tests. Moreover, the Branch & Bound with the subgradient method embedded explored very few nodes to reach the optimal value, if compared with an exact solver that used the linear relaxation. However, before proposing an exact algorithm based on our Lagrangian relaxation some improvements to the implemented Branch & Bound must be considered.

Future works will consider combining our subgradient method with some metaheuristics to improve the results and to test on larger instances, similar strategies were successfully applied for other problems such as the multi-activity shift scheduling problem (Hernández-Leandro et al., 2019), the maximum edge disjoint path problem (Weiner et al., 2021), and the aircraft maintenance routing problem (Bulbul and Kasimbeyli, 2021). Another possible direction would be to formally study the quality of the mathematical models and to analyze the existence of stronger formulations. Also, the applicability of these techniques should be studied for related problems in viral marketing as influence maximization with deactivation in social networks (Tanınmış et al., 2019), and seeding of new products (Negahban and Smith, 2018).

CRediT authorship contribution statement

Santiago V. Ravelo: Conceptualization, Methodology, Software, Validation, Writing - original draft, Writing - review & editing. **Cláudio N. Meneses:** Methodology, Validation, Writing - original draft, Writing - review & editing.

Acknowledgments

This work was partially supported by CNPq (grant 312206/2015-1).

Appendix. Experiments results

This appendix contains tables with the values obtained in the experiments, used to generate the figures that supported the results analysis of the computational tests.

Table A.3
Results of the subgradient method bounds in comparison with the linear relaxation for the MIN-ER-TSS. The execution times are given in seconds. The best bounds for each instance are highlighted with **bold text**.

Dataset	Subgradient		Linear relaxation		Dataset	Subgradient		Linear relaxation	
	Bound	Time	Bound	Time		Bound	Time	Bound	Time
1	3.0	<0.1	**3.6**	0.2	42	**4.0**	0.7	0.0	0.4
2	**2.0**	<0.1	0.0	0.1	43	**6.0**	0.7	0.0	0.4
3	**10.0**	0.1	0.0	0.2	44	**1.0**	0.7	0.0	0.4
4	**5.0**	0.1	0.0	0.2	45	**5.0**	0.7	0.0	0.4
5	**4.0**	0.1	0.0	0.2	46	**6.0**	0.6	0.1	0.4
6	**6.0**	0.2	0.0	0.2	47	**6.0**	0.7	0.0	0.3
7	**5.0**	0.2	0.0	0.2	48	**5.0**	0.7	0.0	0.5
8	**23.0**	4.4	22.8	0.3	49	**5.0**	0.7	0.0	0.4
9	**5.0**	0.2	0.0	0.2	50	**5.0**	0.7	0.0	0.4
10	**15.0**	0.3	0.0	0.3	51	**9.0**	0.7	0.0	0.4
11	**15.0**	0.3	0.0	0.4	52	**6.0**	0.6	0.0	0.3
12	**15.0**	0.3	0.0	0.3	53	**7.0**	0.6	0.0	0.4
13	**15.0**	0.3	0.0	0.3	54	7.0	0.6	**13.8**	0.3
14	**14.0**	0.3	0.0	0.3	55	**7.0**	0.6	0.0	0.4
15	**13.0**	0.3	0.0	0.3	56	**5.0**	0.7	0.0	0.4
16	**14.0**	0.3	0.0	0.3	57	**7.0**	0.6	0.0	0.3
17	**14.0**	0.3	0.0	0.3	58	**6.0**	0.7	0.0	0.4
18	**14.0**	0.3	0.0	0.3	59	7.0	0.6	**8.9**	0.4
19	**13.0**	0.3	0.0	0.3	60	7.0	0.6	**9.5**	0.3
20	**13.0**	0.3	0.0	0.4	61	**7.0**	0.6	0.0	0.3
21	**13.0**	0.3	0.0	0.4	62	**6.0**	0.7	0.0	0.4
22	**13.0**	0.3	0.0	0.3	63	**7.0**	0.6	0.2	0.4
23	**13.0**	0.3	0.0	0.3	64	**7.0**	0.7	0.0	0.4
24	**13.0**	0.3	0.0	0.4	65	**4.0**	0.7	0.0	0.3
25	**11.0**	0.4	0.0	0.4	66	**7.0**	0.6	0.0	0.3
26	**6.0**	0.4	0.0	0.3	67	**7.0**	0.7	0.0	0.3
27	**6.0**	0.4	0.0	0.3	68	**4.0**	0.6	0.0	0.4
28	**5.0**	0.4	0.0	0.3	69	**7.0**	0.6	0.0	0.3
29	**6.0**	0.4	0.0	0.3	70	**6.0**	0.7	0.0	0.4
30	**4.0**	0.4	0.0	0.3	71	**7.0**	0.6	4.3	0.3
31	**11.0**	0.6	0.0	0.4	72	**7.0**	0.7	0.0	0.3
32	**5.0**	0.7	0.0	0.4	73	**11.0**	3.7	0.0	1.2
33	**9.0**	0.7	0.0	0.4	74	**11.0**	3.6	0.0	1.2
34	**8.0**	0.7	0.0	0.4	75	**26.0**	3.5	0.0	2.1
35	**11.0**	0.7	0.0	0.4	76	**12.0**	5.3	0.0	1.8
36	**7.0**	0.7	0.0	0.4	77	**12.0**	5.3	0.0	2.0
37	**9.0**	0.7	0.0	0.4	78	**10.0**	5.3	0.0	1.5
38	**10.0**	0.7	0.0	0.4	79	**8.0**	5.4	0.0	1.8
39	**11.0**	0.7	0.0	0.5	80	**60.0**	5.7	0.0	3.5
40	**11.0**	0.7	0.0	0.4	81	**7.0**	7.6	0.0	3.2
41	**6.0**	0.6	0.0	0.4	82	**30.0**	16.5	0.0	11.3

Table A.4
Results of the subgradient method bounds in comparison with the linear relaxation for the MAX-ER-TSS. The execution times are given in seconds. The best bounds for each instance are highlighted with **bold text**.

Dataset	Subgradient		Linear relaxation		Dataset	Subgradient		Linear relaxation	
	Bound	Time	Bound	Time		Bound	Time	Bound	Time
1	34.0	<0.1	**26.3**	0.1	42	**103.0**	<0.001	**103.0**	0.4
2	**46.0**	<0.001	**46.0**	0.1	43	**100.0**	<0.001	**100.0**	0.4
3	**63.0**	0.1	**63.0**	0.2	44	**103.0**	0.7	**103.0**	0.4
4	49.0	0.2	**46.2**	0.2	45	**99.0**	<0.1	**99.0**	0.4
5	49.0	0.2	**46.7**	0.2	46	**99.0**	<0.1	**99.0**	0.3
6	**62.0**	0.2	**62.0**	0.2	47	**87.0**	<0.1	**87.0**	0.4
7	**69.0**	<0.1	**69.0**	0.2	48	**103.0**	<0.1	**103.0**	0.5
8	32.0	0.3	**9.6**	0.2	49	**106.0**	0.7	**106.0**	0.4
9	**69.0**	0.2	**69.0**	0.3	50	**98.0**	0.7	**98.0**	0.4
10	**14.0**	0.3	85.0	0.4	51	**104.0**	0.7	**104.0**	0.4
11	**14.0**	0.3	85.0	0.3	52	87.0	0.7	**85.0**	0.4
12	**13.0**	0.4	85.0	0.3	53	57.0	0.7	**51.0**	0.4
13	**14.0**	0.3	86.0	0.4	54	32.0	0.7	**16.6**	0.3
14	**14.0**	0.3	86.0	0.4	55	65.0	0.7	**54.1**	0.4
15	**21.0**	0.3	84.0	0.3	56	88.0	0.7	**87.9**	0.4
16	**14.0**	0.4	85.0	0.3	57	65.0	0.7	**58.9**	0.5
17	**14.0**	0.3	86.0	0.3	58	89.0	0.7	**87.8**	0.3
18	**13.0**	0.3	85.0	0.3	59	31.0	0.7	**16.9**	0.3
19	**20.0**	0.3	84.0	0.4	60	33.0	0.7	**19.2**	0.3
20	**21.0**	0.3	84.0	0.3	61	79.0	0.7	**76.0**	0.4
21	**20.0**	0.3	84.0	0.3	62	**96.0**	0.7	**96.0**	0.5
22	**20.0**	0.3	84.0	0.3	63	58.0	0.7	**46.6**	0.3
23	**20.0**	0.3	84.0	0.3	64	**82.0**	<0.001	**82.0**	0.4
24	**22.0**	0.3	87.0	0.3	65	74.0	0.7	**67.6**	0.7
25	**13.0**	0.4	88.0	0.4	66	59.0	0.7	**52.0**	0.4
26	**92.0**	<0.001	**92.0**	0.2	67	64.0	0.7	**51.2**	0.5
27	**89.0**	<0.1	**89.0**	0.3	68	58.0	0.7	**51.9**	0.5
28	**91.0**	0.4	**91.0**	0.4	69	63.0	0.7	**58.9**	0.4
29	**86.0**	<0.001	**86.0**	0.4	70	**92.0**	0.7	**90.8**	0.4
30	**74.0**	0.4	**74.0**	0.3	71	41.0	0.7	**28.8**	0.4
31	**22.0**	0.7	103.0	0.4	72	75.0	0.7	**71.0**	0.4
32	**106.0**	<0.1	**106.0**	0.4	73	**173.0**	3.8	**173.0**	1.1
33	**104.0**	0.7	**104.0**	0.4	74	**161.0**	3.8	**161.0**	1.1
34	**105.0**	0.7	**105.0**	0.4	75	**11.0**	<0.001	171.0	1.8
35	**102.0**	0.7	**102.0**	0.4	76	165.0	5.7	**160.9**	1.4
36	**102.0**	<0.1	**102.0**	0.3	77	182.0	5.6	**181.9**	1.7
37	**105.0**	0.7	**105.0**	0.4	78	**188.0**	5.5	**188.0**	1.6
38	**104.0**	0.7	**104.0**	0.4	79	**197.0**	5.5	**197.0**	1.6
39	**106.0**	0.8	**106.0**	0.4	80	**12.0**	5.7	198.0	2.8
40	**107.0**	<0.1	**107.0**	0.4	81	211.0	7.7	**210.1**	2.1
41	**101.0**	<0.1	**101.0**	0.3	82	**24.0**	16.7	288.0	7.9

Table A.5
Comparison between subgradient and linear relaxation bounds in B&B executions for the MIN-ER-TSS. The column `L.R. % S.` gives the average values of the linear relaxation bound as a percent of the subgradient method bound at each node of the B&B enumeration tree. The columns `Time S.` and `Time L.R.` are, respectively, the average time in seconds of the subgradient method and the linear relaxation calls at each node of the B&B execution.

Dataset	L.R. % S.	Time S.	Time L.R.	Dataset	L.R. % S.	Time S.	Time L.R.
1	97.8	<0.1	0.1	42	0.0	0.1	0.3
2	0.6	<0.1	0.2	43	2.6	0.1	0.3
3	0.0	0.1	0.2	44	59.6	0.1	0.3
4	5.3	<0.1	0.2	45	0.0	0.1	0.3
5	5.2	<0.1	0.2	46	4.9	0.1	0.3
6	2.7	<0.1	0.2	47	0.0	0.2	0.3
7	0.0	<0.1	0.2	48	0.0	0.2	0.5
8	417.8	<0.1	0.2	49	0.0	0.2	0.4
9	1.4	0.1	0.2	50	0.0	0.2	0.4
10	0.0	0.1	0.2	51	0.0	0.1	0.4
11	0.0	<0.1	0.2	52	1.8	0.2	0.4
12	0.0	<0.1	0.2	53	3.1	0.2	0.4
13	0.0	0.1	0.2	54	190.5	0.1	0.3
14	0.0	0.1	0.2	55	1.5	0.1	0.3
15	0.0	<0.1	0.2	56	0.0	0.1	0.3
16	0.0	0.1	0.2	57	3.2	0.1	0.3
17	0.0	<0.1	0.2	58	2.1	0.1	0.3
18	0.0	0.1	0.2	59	94.5	0.1	0.3
19	0.0	<0.1	0.2	60	127.7	0.2	0.2
20	0.0	0.1	0.2	61	4.5	0.1	0.3
21	0.0	<0.1	0.2	62	1.3	0.1	0.3
22	0.0	0.1	0.2	63	1.8	0.1	0.3
23	0.0	<0.1	0.2	64	11.1	0.1	0.3
24	0.0	<0.1	0.2	65	4.5	0.1	0.3
25	1.2	0.1	0.3	66	4.2	0.1	0.3
26	0.0	0.1	0.3	67	1.4	0.1	0.3
27	9.1	0.1	0.3	68	3.0	0.1	0.3
28	4.8	0.1	0.2	69	4.0	0.1	0.2
29	0.0	0.1	0.3	70	3.0	0.1	0.3
30	7.0	0.1	0.2	71	65.6	0.8	0.2
31	0.0	0.1	0.3	72	4.9	0.1	0.3
32	0.0	0.1	0.3	73	0.0	0.6	1.2
33	0.0	0.1	0.3	74	0.0	0.6	1.3
34	0.0	0.1	0.3	75	0.0	0.3	0.9
35	0.0	0.1	0.3	76	1.5	0.8	1.2
36	0.0	0.1	0.3	77	1.4	0.8	1.2
37	0.0	0.1	0.3	78	0.0	0.5	1.0
38	0.0	0.1	0.3	79	0.0	0.6	1.0
39	0.0	0.1	0.3	80	0.0	0.4	1.1
40	0.0	0.1	0.3	81	0.7	0.6	1.2
41	0.0	0.1	0.3	82	0.0	0.7	1.9

Table A.6
Comparison between subgradient and linear relaxation bounds in B&B executions for the MAX-ER-TSS. The column `L.R. % S.` gives the average values of the linear relaxation bound as a percent of the subgradient method bound at each node of the B&B enumeration tree. The columns `Time S.` and `Time L.R.` are, respectively, the average time in seconds of the subgradient method and the linear relaxation calls at each node of the B&B execution.

Dataset	L.R. % S.	Time S.	Time L.R.	Dataset	L.R. % S.	Time S.	Time L.R.
1	107.9	<0.1	0.2	42	100.0	<0.1	0.4
2	100.0	<0.001	0.2	43	100.0	<0.1	0.4
3	211.3	<0.1	0.2	44	1834.6	0.4	0.4
4	807.9	<0.1	0.2	45	100.0	0.1	0.3
5	467.5	<0.1	0.2	46	100.0	<0.1	0.3
6	98.9	<0.1	0.2	47	100.0	<0.001	0.3
7	100.0	<0.1	0.2	48	100.0	0.1	0.4
8	67.1	0.1	0.3	49	3191.3	0.4	0.4
9	1270.3	0.1	0.2	50	2034.9	0.4	0.3
10	1084.2	0.2	0.3	51	2065.3	0.4	0.4
11	1106.5	0.2	0.3	52	1357.5	0.3	0.3
12	1087.0	0.2	0.3	53	776.7	0.3	0.3
13	1122.1	0.2	0.3	54	108.5	0.3	0.3
14	1370.7	0.2	0.3	55	696.6	0.3	0.3
15	1597.7	0.2	0.3	56	1453.1	0.4	0.3
16	1329.8	0.2	0.3	57	950.3	0.3	0.3
17	1350.6	0.2	0.3	58	1635.8	0.4	0.3
18	1342.0	0.2	0.3	59	120.6	0.3	0.3
19	1685.2	0.2	0.3	60	118.5	0.3	0.3
20	1709.8	0.2	0.3	61	682.0	0.3	0.3
21	1835.8	0.2	0.3	62	1961.6	0.3	0.3
22	1835.8	0.2	0.3	63	533.0	0.3	0.3
23	1673.3	0.2	0.3	64	100.0	<0.1	0.3
24	1730.3	0.2	0.3	65	1008.6	0.4	0.3
25	1746.6	0.3	0.3	66	944.9	0.3	0.3
26	100.0	<0.001	0.3	67	625.8	0.3	0.3
27	100.0	0.1	0.3	68	621.9	0.3	0.3
28	100.0	0.2	0.3	69	910.6	0.3	0.3
29	100.0	<0.001	0.3	70	1634.1	0.3	0.3
30	1671.2	0.2	0.3	71	157.3	0.3	0.3
31	2414.5	0.4	0.4	72	1314.3	0.3	0.3
32	100.0	<0.1	0.4	73	3669.9	1.7	0.9
33	1431.6	0.4	0.4	74	2363.4	2.0	0.9
34	1936.9	0.5	0.4	75	1554.5	<0.1	1.9
35	2146.5	0.4	0.4	76	3693.6	2.9	1.2
36	100.0	<0.1	0.4	77	4010.7	3.0	1.3
37	2129.8	0.4	0.4	78	4890.3	3.0	1.4
38	3752.1	0.3	0.4	79	4693.3	3.1	1.5
39	1509.7	0.4	0.4	80	1650.0	5.6	2.7
40	100.0	<0.1	0.4	81	2932.2	5.7	2.1
41	100.0	<0.1	0.3	82	4048.0	14.2	7.4

Table A.7
Subgradient method, B&B using subgradient method and Gurobi solver comparison for the MIN-ER-TSS. The column `Sol.` gives the best solution value found by the approach. The column `GAP` gives the GAP between the solution value and the best bound found by the approach. The column `Time` gives the approach execution time in seconds. The column `Expl.` gives the number of nodes explored by the enumerative approaches.

Dataset	Subgradient			B & B				Gurobi			
	Sol.	GAP	Time	Sol.	GAP	Time	Expl.	Sol.	GAP	Time	Expl.
1	6[a]	50.0	0.0	6[a]	0.0	0.4	28	6[a]	0.0	0.1	5
2	4[a]	50.0	0.0	4[a]	0.0	0.4	33	4[a]	0.0	0.3	99
3	10[a]	0.0	0.1	10[a]	0.0	0.1	1	10[a]	0.0	0.6	1 394
4	11[a]	54.5	0.1	11[a]	0.0	2.7	85	11[a]	0.0	0.4	180
5	5[a]	20.0	0.1	5[a]	0.0	0.5	17	5[a]	0.0	0.4	19
6	12[a]	50.0	0.2	12[a]	0.0	2.8	75	12[a]	0.0	0.5	901
7	11[a]	54.5	0.2	11[a]	0.0	2.1	49	11[a]	0.0	1.0	5 377
8	26[a]	12.5	4.4	26[a]	0.0	41.3	1427	26[a]	0.0	0.4	213
9	12	58.3	0.2	11[a]	0.0	6.3	156	11[a]	0.0	0.7	2 949
10	18	16.6	0.3	16[a]	0.0	39.9	963	16[a]	0.0	41.2	87 866
11	19	21.1	0.3	17[a]	0.0	66.4	1760	17[a]	0.0	61.2	125 085
12	18	16.6	0.3	17[a]	0.0	52.9	1310	17[a]	0.0	45.9	74 156
13	19	21.1	0.3	16[a]	0.0	26.3	627	16[a]	0.0	28.3	48 359
14	18	22.2	0.3	17[a]	0.0	44.6	1066	17[a]	0.0	66.6	91 359
15	19	31.6	0.3	17[a]	0.0	60.5	1488	17[a]	0.0	32.3	58 811
16	19	26.3	0.3	17[a]	0.0	43.6	1041	17[a]	0.0	56.0	104 899
17	19	26.3	0.3	17[a]	0.0	39.6	899	17[a]	0.0	64.6	116 626
18	21	33.3	0.3	17[a]	0.0	46.9	1128	17[a]	0.0	80.8	119 055
19	21	38.1	0.3	17[a]	0.0	71.6	1738	17[a]	0.0	68.0	108 463
20	18	27.8	0.3	16[a]	0.0	33.1	719	16[a]	0.0	61.8	91 712
21	20	35.0	0.3	17[a]	0.0	53.2	1280	17[a]	0.0	66.1	130 069
22	19	31.6	0.3	16[a]	0.0	48.2	1212	16[a]	0.0	50.8	79 409
23	21	38.1	0.3	17[a]	0.0	48.2	1129	17[a]	0.0	76.7	115 327
24	19	31.6	0.3	18[a]	0.0	56.8	1314	18[a]	0.0	39.3	57 824
25	18	38.9	0.4	14[a]	0.0	45.2	930	14[a]	0.0	32.5	44 485
26	9	33.3	0.4	8[a]	0.0	3.0	39	8[a]	0.0	0.5	264
27	8	25.0	0.4	7[a]	0.0	4.2	57	7[a]	0.0	0.5	23
28	6[a]	16.7	0.4	6[a]	0.0	1.6	19	6[a]	0.0	0.6	46
29	7[a]	14.3	0.4	7[a]	0.0	2.7	31	7[a]	0.0	0.5	13
30	14[a]	71.4	0.4	14[a]	0.0	5.6	135	14[a]	0.0	0.6	707
31	24	54.2	0.6	14[a]	0.0	86.2	893	14[a]	0.0	51.5	58 360
32	11[a]	54.5	0.7	11[a]	0.0	12.7	101	11[a]	0.0	0.6	38
33	17[a]	47.1	0.7	17[a]	0.0	184.3	2008	17[a]	0.0	16.0	29 244
34	18	55.6	0.7	17[a]	0.0	88.3	837	17[a]	0.0	23.6	36 503
35	20[a]	45.0	0.6	20[a]	0.0	62.9	648	20[a]	0.0	9.4	19 557
36	14[a]	50.0	0.7	14[a]	0.0	3.8	31	14[a]	0.0	0.5	11
37	20	55.0	0.7	16[a]	0.0	51.6	518	16[a]	0.0	3.8	8 205
38	19	47.4	0.7	15[a]	0.0	60.5	599	15[a]	0.0	1.1	2 340
39	21	47.6	0.7	20[a]	0.0	148.6	1639	20[a]	0.0	13.6	21 189
40	13	15.4	0.7	11[a]	0.0	8.9	119	11[a]	0.0	5.5	5 705
41	12[a]	50.0	0.6	12[a]	0.0	8.9	87	12[a]	0.0	0.7	959
42	6[a]	33.3	0.7	6[a]	0.0	8.3	77	6[a]	0.0	1.2	1 350
43	9[a]	33.3	0.7	9[a]	0.0	3.9	33	9[a]	0.0	0.5	7
44	18	94.4	0.7	15[a]	0.0	105.0	1012	15[a]	0.0	0.7	611
45	11[a]	54.5	0.7	11[a]	0.0	5.3	49	11[a]	0.0	1.6	4 529
46	11[a]	45.5	0.6	11[a]	0.0	5.2	41	11[a]	0.0	0.6	8
47	9[a]	33.3	0.7	9[a]	0.0	5.4	41	9[a]	0.0	0.4	6
48	11[a]	54.5	0.7	11[a]	0.0	40.2	428	11[a]	0.0	4.9	5 202
49	17[a]	70.6	0.7	17[a]	0.0	57.6	621	17[a]	0.0	0.9	1 996
50	17	70.6	0.7	16[a]	0.0	24.2	233	16[a]	0.0	1.4	2 565
51	26	65.4	0.7	24[a]	0.0	276.5	3232	24[a]	0.0	20.9	31 686

[a]Used to identify optimal values.

Table A.8
Subgradient method, B&B using subgradient method and Gurobi solver comparison for the MIN-ER-TSS. The column `Sol.` gives the best solution value found by the approach. The column `GAP` gives the GAP between the solution value and the best bound found by the approach. The column `Time` gives the approach execution time in seconds. The column `Expl.` gives the number of nodes explored by the enumerative approaches.

Dataset	Subgradient			B & B				Gurobi			
	Sol.	GAP	Time	Sol.	GAP	Time	Expl.	Sol.	GAP	Time	Expl.
52	8[a]	25.0	0.6	8[a]	0.0	6.4	55	8[a]	0.0	0.9	2 114
53	9[a]	22.2	0.6	9[a]	0.0	8.9	71	9[a]	0.0	0.5	14
54	32	78.1	0.6	25[a]	0.0	134.8	1447	25[a]	0.0	0.4	3
55	29	75.9	0.6	16[a]	0.0	35.6	326	16[a]	0.0	0.5	396
56	18	72.2	0.7	15[a]	0.0	30.4	279	15[a]	0.0	0.6	462
57	8[a]	12.5	0.6	8[a]	0.0	4.3	35	8[a]	0.0	0.5	23
58	17	64.7	0.7	16[a]	0.0	36.9	323	16[a]	0.0	1.2	4 377
59	17[a]	58.8	0.6	17[a]	0.0	21.1	191	17[a]	0.0	0.4	1
60	24[a]	70.8	0.6	24[a]	0.0	152.7	1625	24[a]	0.0	0.4	1
61	16[a]	56.2	0.6	16[a]	0.0	22.3	205	16[a]	0.0	0.7	445
62	16[a]	62.5	0.7	16[a]	0.0	31.1	304	16[a]	0.0	1.3	4 097
63	22[a]	68.2	0.6	22[a]	0.0	39.9	376	22[a]	0.0	0.6	882
64	9[a]	22.2	0.7	9[a]	0.0	4.8	45	9[a]	0.0	0.4	19
65	8[a]	50.0	0.7	8[a]	0.0	8.4	75	8[a]	0.0	0.5	312
66	16	56.3	0.6	15[a]	0.0	18.9	173	15[a]	0.0	0.6	476
67	9[a]	22.2	0.7	9[a]	0.0	7.1	63	9[a]	0.0	0.5	28
68	12[a]	66.7	0.6	12[a]	0.0	12.0	123	12[a]	0.0	0.5	114
69	16[a]	56.3	0.6	16[a]	0.0	23.6	226	16[a]	0.0	0.5	26
70	15[a]	60.0	0.7	15[a]	0.0	46.6	416	15[a]	0.0	1.0	2 225
71	10[a]	30.0	0.6	10[a]	0.0	9.0	79	10[a]	0.0	0.4	3
72	8[a]	12.5	0.7	8[a]	0.0	5.1	41	8[a]	0.0	0.5	402
73	26	57.7	3.7	25[a]	0.0	452.5	903	25[a]	0.0	116.8	81 188
74	25[a]	56.0	3.6	25[a]	0.0	635.7	1335	25[a]	0.0	47.4	37 360
75	50	48.0	3.5	44	40.9	3600.0	3599	–	–	3600.0	123 889
76	15[a]	20.0	5.3	15[a]	0.0	111.2	149	15[a]	0.0	22.1	5 722
77	14[a]	16.7	5.3	14[a]	0.0	129.6	171	14[a]	0.0	44.7	6 681
78	68	85.3	5.3	56	82.1	3600.0	1234	122	99.2	3600.0	306 980
79	75	89.3	5.4	60	86.7	3600.0	1355	–	–	3600.0	190 541
80	76	21.1	5.7	66	9.1	3600.0	1573	–	–	3600.0	50 545
81	54	87.0	7.6	53	86.8	3600.0	741	–	–	3600.0	85 080
82	101	70.3	16.5	82	63.4	3600.0	5394	–	–	3600.0	7 224

[a]Used to identify optimal values.

Table A.9
Subgradient method, B&B using subgradient method and Gurobi solver comparison for the MAX-ER-TSS. The column `Sol.` gives the best solution value found by the approach. The column `GAP` gives the GAP between the solution value and the best bound found by the approach. The column `Time` gives the approach execution time in seconds. The column `Expl.` gives the number of nodes explored by the enumerative approaches.

Dataset	Subgradient			B & B				Gurobi			
	Sol.	GAP	Time	Sol.	GAP	Time	Expl.	Sol.	GAP	Time	Expl.
1	21[a]	61.9	<0.1	21[a]	0.0	0.5	79	21[a]	0.0	0.1	9
2	46[a]	0.0	<0.001	46[a]	0.0	<0.1	1	46[a]	0.0	0.2	0
3	24	162.5	0.1	37[a]	0.0	2.9	199	37[a]	0.0	0.9	4 529
4	23	113.0	0.2	44[a]	0.0	3.7	199	44[a]	0.0	0.4	213
5	44	11.4	0.2	45[a]	0.0	2.4	137	45[a]	0.0	0.5	38
6	60[a]	3.3	0.2	60[a]	0.0	4.8	229	60[a]	0.0	0.7	2 141
7	69[a]	0.0	<0.1	69[a]	0.0	<0.1	1	69[a]	0.0	0.3	0
8	8[a]	300.0	0.3	8[a]	0.0	6.6	305	8[a]	0.0	0.4	12
9	18	283.3	0.2	57[a]	0.0	5.3	231	57[a]	0.0	0.7	1 708
10	10	40.0	0.3	12[a]	0.0	3.2	89	12[a]	0.0	3	11 121
11	10	40.0	0.3	12[a]	0.0	3.1	85	12[a]	0.0	2.8	7 271
12	11	18.2	0.4	12[a]	0.0	3.2	89	12[a]	0.0	4.2	11 592
13	10	40.0	0.3	12[a]	0.0	3.1	85	12[a]	0.0	3.5	11 107
14	12[a]	16.7	0.3	12[a]	0.0	3.7	105	12[a]	0.0	2.2	4 392
15	11	90.9	0.3	13[a]	0.0	9.0	269	13[a]	0.0	3.9	13 907
16	10	40.0	0.4	12[a]	0.0	5.2	147	12[a]	0.0	1.8	9 706
17	12[a]	16.7	0.3	12[a]	0.0	5.2	147	12[a]	0.0	3.4	14 493
18	11	18.2	0.3	12[a]	0.0	5.2	147	12[a]	0.0	2.2	5 672
19	10	100.0	0.3	12[a]	0.0	8.9	269	12[a]	0.0	3.2	10 871
20	10	110.0	0.3	12[a]	0.0	8.8	269	12[a]	0.0	2.3	8 639
21	11	81.8	0.3	12[a]	0.0	8.7	259	12[a]	0.0	1.9	10 374
22	11	81.8	0.3	12[a]	0.0	8.9	271	12[a]	0.0	3.2	19 859
23	12[a]	66.7	0.3	12[a]	0.0	8.7	259	12[a]	0.0	1.3	5 961
24	12	83.3	0.3	13[a]	0.0	8.8	257	13[a]	0.0	0.8	788
25	12	8.3	0.4	13[a]	0.0	0.3	4	13[a]	0.0	1.9	7 805
26	92[a]	0.0	<0.001	92[a]	0.0	<0.001	1	92[a]	0.0	0.5	221
27	89[a]	0.0	<0.1	89[a]	0.0	<0.1	1	89[a]	0.0	0.5	11
28	58	56.9	0.4	91[a]	0.0	0.2	3	91[a]	0.0	0.4	11
29	86[a]	0.0	<0.1	86[a]	0.0	<0.1	1	86[a]	0.0	0.5	109
30	43[a]	72.1	0.4	43[a]	0.0	11.8	295	43[a]	0.0	0.5	153
31	13	69.2	0.7	16[a]	0.0	8.2	109	16[a]	0.0	2.6	9 613
32	106[a]	0.0	<0.1	106[a]	0.0	<0.1	1	106[a]	0.0	0.7	83
33	15	593.3	0.7	21[a]	0.0	15.2	217	21[a]	0.0	3.2	10 894
34	20[a]	425.0	0.7	20[a]	0.0	15.7	207	20[a]	0.0	8.8	10 473
35	14	628.6	0.7	15[a]	0.0	33.5	503	15[a]	0.0	2.3	17 206
36	102[a]	0.0	<0.1	102[a]	0.0	<0.1	1	102[a]	0.0	0.6	30
37	15	600.0	0.7	21[a]	0.0	11.4	137	21[a]	0.0	1.5	2 454
38	14	642.9	0.7	15[a]	0.0	25.0	373	15[a]	0.0	1.4	2 573
39	15	606.7	0.8	35[a]	0.0	18.7	263	35[a]	0.0	7.4	12 101
40	107[a]	0.0	<0.1	107[a]	0.0	<0.1	1	107[a]	0.0	3.8	10 365
41	101[a]	0.0	<0.1	101[a]	0.0	<0.1	1	101[a]	0.0	0.6	31
42	103[a]	0.0	<0.001	103[a]	0.0	<0.1	1	103[a]	0.0	0.5	0
43	100[a]	0.0	<0.001	100[a]	0.0	<0.1	1	100[a]	0.0	0.5	1
44	23[a]	347.8	0.7	23[a]	0.0	75.9	1313	23[a]	0.0	0.7	75
45	99[a]	0.0	<0.1	99[a]	0.0	<0.1	1	99[a]	0.0	0.5	0
46	99[a]	0.0	<0.1	99[a]	0.0	<0.1	1	99[a]	0.0	0.6	1
47	87[a]	0.0	<0.1	87[a]	0.0	<0.1	1	87[a]	0.0	0.5	7
48	103[a]	0.0	<0.1	103[a]	0.0	<0.1	1	103[a]	0.0	0.7	386
49	18[a]	488.9	0.7	18[a]	0.0	24.9	385	18[a]	0.0	0.9	847
50	21[a]	366.7	0.7	21[a]	0.0	26.3	417	21[a]	0.0	0.8	591
51	14	642.9	0.7	16[a]	0.0	38.9	583	16[a]	0.0	4.8	11 130

[a]Used to identify optimal values.

Table A.10
Subgradient method, B&B using subgradient method and Gurobi solver comparison for the Max-ER-TSS. The column `Sol.` gives the best solution value found by the approach. The column `GAP` gives the GAP between the solution value and the best bound found by the approach. The column `Time` gives the approach execution time in seconds. The column `Expl.` gives the number of nodes explored by the enumerative approaches.

Dataset	Subgradient			B & B				Gurobi			
	Sol.	GAP	Time	Sol.	GAP	Time	Expl.	Sol.	GAP	Time	Expl.
52	83[a]	4.8	0.7	83[a]	0.0	25.3	419	83[a]	0.0	2.4	6 067
53	44[a]	29.5	0.7	44[a]	0.0	21.7	337	44[a]	0.0	0.6	92
54	9	255.5	0.7	11[a]	0.0	29.4	447	11[a]	0.0	0.5	28
55	22	195.5	0.7	33[a]	0.0	27.2	451	33[a]	0.0	0.7	515
56	40	120.0	0.7	47[a]	0.0	25.5	409	47[a]	0.0	0.9	616
57	47[a]	38.3	0.7	47[a]	0.0	22.3	371	47[a]	0.0	0.6	107
58	28[a]	217.9	0.7	28[a]	0.0	26.4	405	28[a]	0.0	1.1	1 905
59	13[a]	138.5	0.7	13[a]	0.0	26.6	411	13[a]	0.0	0.6	3
60	10	230.0	0.7	12[a]	0.0	28.6	457	12[a]	0.0	0.5	1
61	74[a]	6.8	0.7	74[a]	0.0	21.0	379	74[a]	0.0	0.8	730
62	46[a]	108.5	0.7	46[a]	0.0	20.5	331	46[a]	0.0	0.8	591
63	26	123.1	0.7	28[a]	0.0	27.4	441	28[a]	0.0	0.6	676
64	82[a]	0.0	<0.001	82[a]	0.0	<0.1	1	82[a]	0.0	0.5	0
65	62[a]	19.4	0.7	62[a]	0.0	19.4	309	62[a]	0.0	0.7	281
66	42[a]	40.5	0.7	42[a]	0.0	28.4	451	42[a]	0.0	0.9	2 916
67	37	73.0	0.7	44[a]	0.0	20.6	389	44[a]	0.0	0.7	190
68	39[a]	48.7	0.7	39[a]	0.0	26.3	435	39[a]	0.0	0.7	545
69	35[a]	80.0	0.7	35[a]	0.0	19.7	351	35[a]	0.0	0.6	21
70	54[a]	70.4	0.7	54[a]	0.0	23.4	373	54[a]	0.0	0.9	1 293
71	28[a]	46.4	0.7	28[a]	0.0	25.7	425	28[a]	0.0	0.6	32
72	69	8.7	0.7	70[a]	0.0	22.3	369	70[a]	0.0	0.8	563
73	73[a]	137.0	3.8	73[a]	0.0	278.1	1049	73[a]	0.0	28.6	15 248
74	27[a]	496.3	3.8	27[a]	0.0	271.3	1009	27[a]	0.0	9.1	5 707
75	11[a]	0.0	<0.001	11[a]	0.0	<0.1	1	11[a]	0.0	26.5	6 099
76	155[a]	6.5	5.7	155[a]	0.0	552.0	1341	155[a]	0.0	14.9	5 408
77	99	83.8	5.6	169[a]	0.0	531.6	1369	169[a]	0.0	19.5	5 988
78	22	754.5	5.5	30[a]	0.0	626.9	1515	23	717.4	3600.0	821 375
79	22	795.5	5.5	23[a]	0.0	512.0	1235	20	885.0	3600.0	465 819
80	10	20.0	5.7	12[a]	0.0	37.7	45	12[a]	0.0	44.6	8 583
81	28	653.6	7.7	44[a]	0.0	1221.4	2025	44[a]	0.0	529.4	16 381
82	14	71.4	16.7	20[a]	20.0	3600.0	2471	20[a]	0.0	807.4	17 034

[a]Used to identify optimal values.

References

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

Baghbani, F.G., Asadpour, M., Faili, H., 2019. Integer linear programming for influence maximization. Iran. J. Sci. Technol. Trans. Electr. Eng. 43 (3), 627–634.

Batagelj, V., Mrvar, A., 2006. Ucinet iv datasets. http://vlado.fmf.uni-lj.si/pub/networks/data/UciNet/UciData.htm (Accessed: 2020-07-18).

Ben-Zwi, O., Hermelin, D., Lokshtanov, D., Newman, I., 2011. Treewidth governs the complexity of target set selection. Discrete Optim. 8 (1), 87–96, Parameterized Complexity of Discrete Optimization.

Bliznets, I., Sagunov, D., 2019. Solving target set selection with bounded thresholds faster than 2^n. In: 13th International Symposium on Parameterized and Exact Computation, Vol. 115. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, pp. 22:1–22:14.

Bulbul, K.G., Kasimbeyli, R., 2021. Augmented Lagrangian based hybrid subgradient method for solving aircraft maintenance routing problem. Comput. Oper. Res. 132, 105294, URL https://www.sciencedirect.com/science/article/pii/S0305054821000861.

Centeno, C.C., Dourado, M.C., Penso, L.D., Rautenbach, D., Szwarcfiter, J.L., 2011. Irreversible conversion of graphs. Theoret. Comput. Sci. 412 (29), 3693–3700.

Chen, N., 2009. On the approximability of influence in social networks. SIAM J. Discrete Math. 23 (3), 1400–1415.

Chiang, C.-Y., Huang, L.-H., Li, B.-J., Wu, J., Yeh, H.-G., 2013. Some results on the target set selection problem. J. Combin. Optim. 25 (4), 702–715.

Cicalese, F., Cordasco, G., Gargano, L., Milanič, M., Peters, J., Vaccaro, U., 2015. Spread of influence in weighted networks under time and budget constraints. Theoret. Comput. Sci. 586, 40–58.

Cordasco, G., Gargano, L., Mecchia, M., Rescigno, A.A., Vaccaro, U., 2018. Discovering small target sets in social networks: A fast and effective algorithm. Algorithmica 80 (6), 1804–1833.

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

Fardad, M., Kearney, G., 2017. On a linear programming approach to the optimal seeding of cascading failures. In: 2017 IEEE 56th Annual Conference on Decision and Control (CDC). pp. 102–107.

Fisher, M.L., 1981. The Lagrangian relaxation method for solving integer programming problems. Manage. Sci. 27 (1), 1–18.

Gaudioso, M., Gorgone, E., Labbé, M., Rodríguez-Chía, A., 2017. Lagrangian Relaxation for SVM feature selection. Comput. Oper. Res. 87, 137–145, URL https://www.sciencedirect.com/science/article/pii/S0305054817301387.

Gurobi Optimization, L., 2020. Gurobi optimizer reference manual. http://www.gurobi.com (Accessed: 2020-07-18).

Hernández-Leandro, N.A., Boyer, V., Salazar-Aguilar, M.A., Rousseau, L.-M., 2019. A matheuristic based on Lagrangian relaxation for the multi-activity shift scheduling problem. European J. Oper. Res. 272 (3), 859–867, URL https://www.sciencedirect.com/science/article/pii/S0377221718306155.

Kempe, D., Kleinberg, J., Tardos, E., 2003. Maximizing the spread of influence through a social network. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. In: KDD '03, ACM, New York, NY, USA, pp. 137–146.

Lazer, D.M.J., Baum, M.A., Benkler, Y., Berinsky, A.J., Greenhill, K.M., Menczer, F., Metzger, M.J., Nyhan, B., Pennycook, G., Rothschild, D., Schudson, M., Sloman, S.A., Sunstein, C.R., Thorson, E.A., Watts, D.J., Zittrain, J.L., 2018. The science of fake news. Science 359 (6380), 1094–1096.

Martin, R.K., 1999. Large Scale Linear and Integer Optimization: A Unified Approach. Springer.

Narayanan, L., Wu, K., 2020. How to choose friends strategically. Theoret. Comput. Sci. 811, 99–111.

Negahban, A., Smith, J.S., 2018. A joint analysis of production and seeding strategies for new products: an agent-based simulation approach. Ann. Oper. Res. 268 (1), 41–62, URL https://doi.org/10.1007/s10479-016-2389-8.

Nichterlein, A., Niedermeier, R., Uhlmann, J., Weller, M., 2013. On tractable cases of target set selection. Soc. Netw. Anal. Min. 3 (2), 233–256.

Raghavan, S., Zhang, R., 2019. A branch-and-cut approach for the weighted target set selection problem on social networks. INFORMS J. Optim. 1 (4), 304–322.

Ravelo, S.V., Meneses, C.N., Anacleto, E.A., 2020. NP-hardness and evolutionary algorithm over new formulation for a target set selection problem. In: 2020 IEEE Congress on Evolutionary Computation (CEC). pp. 1–8.

Richardson, M., Domingos, P., 2002. Mining knowledge-sharing sites for viral marketing. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. In: KDD '02, ACM, New York, NY, USA, pp. 61–70.

Tanınmış, K., Aras, N., Altınel, I., 2019. Influence maximization with deactivation in social networks. European J. Oper. Res. 278 (1), 105–119, URL https://www.sciencedirect.com/science/article/pii/S0377221719303303.

Weiner, J., Ernst, A.T., Li, X., Sun, Y., Deb, K., 2021. Solving the maximum edge disjoint path problem using a modified Lagrangian particle swarm optimisation hybrid. European J. Oper. Res. 293 (3), 847–862, URL https://www.sciencedirect.com/science/article/pii/S0377221721000114.

Wu, T., Shi, Z., Zhang, C., 2021. The hub location problem with market selection. Comput. Oper. Res. 127, 105136, URL https://www.sciencedirect.com/science/article/pii/S0305054820302537.

Zhao, X., Luh, P.B., Wang, J., 1999. Surrogate gradient algorithm for Lagrangian relaxation. J. Optim. Theory Appl. 100 (3), 699–712.