A Relax-and-Cut Framework for Solving Large-Scale (Constrained) Maximum Weight Connected Subgraph Problems

Eduardo Álvarez-Miranda *[1] and Markus Sinnl†[2]

[1] *Department of Industrial Engineering, Universidad de Talca, Curicó, Chile*
[2] *Department of Statistics and Operations Research, Faculty of Business, Economics and Statistics, University of Vienna, Vienna, Austria*

Abstract

Finding maximum weight connected subgraphs within networks is a fundamental combinatorial optimization problem both from a theoretical and a practical standpoint. One of the most prominent applications of this problem appears in Systems Biology and it corresponds to the detection of *active subnetworks* within gene interaction networks.

Due to its importance, several modeling and algorithmic strategies have been proposed for tackling the maximum weight connected subgraph problem (MWCS) over the last years; the most effective strategies typically depend on the use of integer linear programming (ILP). Nonetheless, this implies that large-scale networks (such as those appearing in Systems Biology) can become burdensome; moreover, not all practitioners may have access to an ILP solver. In this paper, a unified modeling and algorithmic scheme is designed to solve the MWCS and some of its application-oriented variants with cardinality or budget constraints. The proposed framework is based on a general node-based model which is tackled by Relax-and-Cut, i.e., Lagrangian relaxation combined with constraint generation. The approach is enhanced by additional valid inequalities, lifted valid inequalities, primal heuristics and variable-fixing procedures.

Computational results on instances from the literature, as well as on additional large-scale instances, show that the proposed framework is competitive with respect to the existing approaches and it allows to find improved solutions for some unsolved instances from literature. The implemented approach is made available online.

1 Introduction and Motivation

The problem of finding *active subnetworks* has recently received considerable attention from the bionformatics community [see, e.g. Andreotti, 2015, Backes et al., 2011, Dittrich et al., 2008, El-Kebir, 2015, Hatem, 2014, Huang, 2011, Ideker et al., 2002, Yamamoto et al., 2009, and the references therein]. In this problem, one is given a gene interaction network (also known as *interactome*), and the goal is to find active subnetworks associated with a particular biological process (e.g., a cancer). In Dittrich et al. [2008], this problem was formalized as the *maximum weight connected subgraph problem* (MWCS). One is given a graph $G(V, E)$, with node-weights $w_v \in \mathbb{R}$, $\forall v \in V$, and the goal is to find a connected subgraph G' with maximum node-weight. In a biological context, the nodes model genes and a particular node-weights w_v is a *score* that represents the significance of gene v for the biological process under investigation. The scores are typically based on data obtained by DNA-microarray experiments.

*ealvarez@utalca.cl
†markus.sinnl@univie.ac.at

Aside from its importance in bioinformatics, the MWCS appears as a basic optimization problem in wildlife corridor design [Dilkina and Gomes, 2010], forestry planning [Carvajal et al., 2013], object and activity saliency detection [Adluru et al., 2014, Chen and Grauman, 2012, Vijayanarasimhan and Grauman, 2011], wireless network deployment planning [Kuo et al., 2015], among others.

Dittrich et al. [2008] showed that the MWCS can be transformed into the *prize-collecting Steiner tree problem* (PCSTP) and developed an exact integer linear programming (ILP)-based solution approach built on the PCSTP framework of [Ljubić et al., 2006]. After Dittrich et al. [2008], further exact solution approaches based on ILPs have been proposed by [Althaus and Blumenstock, 2014, Álvarez-Miranda et al., 2013a,b, Backes et al., 2011, El-Kebir and Klau, 2014, Fischetti et al., 2014]; in contrast to [Dittrich et al., 2008], where arc- and node-variables are used, these latter approaches are based on formulations using only node-variables. Such models have much less variables which is particularly useful in highly dense networks as those appearing in the identification of functional modules in gene regulatory networks. Complementary to these algorithms, polyhedral studies on the connected subgraph polytope are carried out in [Wang et al., 2014].

Depending on the area of application, there may be additional side-constraints like a cardinality constraint or a budget constraint. In [Backes et al., 2011], the cardinality-constrained counterpart of the MWCS was tackled via integer programming considering an arc-based model; in a latter work, the same variant was approached in [Álvarez-Miranda et al., 2013a] using a much more efficient node-based model. Similarly, an arc-based ILP model was proposed in [Dilkina and Gomes, 2010] for the budget-constrained variant; in a more recent work, considerably better computational results were obtained by means of a node-based model [see Álvarez-Miranda et al., 2013b].

In this work, a Relax-and-Cut (R&C) approach, i.e., Lagrangian relaxation combined with constraint generation, is designed for the MWCS, and its cardinality- and budget-constrained versions. The use of such algorithm is motivated by the following two observations. First, the previously proposed ILP approaches need an exponential number of constraints, hence, they are tackled by means of branch-and-cut. As a consequence, these strategies typically fail in providing acceptable gaps for massive instances, as those appearing from bioinformatics, mainly due to time consuming separations. And second, practitioners may not have access to an ILP solver or may not have the expertise to use it, and thus there is need for alternative approaches. As a matter of fact, for this reason the R-package `BioNet` [see Beisser et al., 2010, in addition to [Dittrich et al., 2008]], also contains a heuristic for the MWCS, for users without access to an ILP solver. Another example of a heuristic for an equivalent problem arising in Bioinformatics corresponds to the hybrid ILP-based heuristic proposed in [Akhmedov et al., 2016]; the approach is based on embedding the resolution of small size PCSTP instances within a clustering strategy in a divide-and-conquer scheme. In contrast to the heuristics in [Akhmedov et al., 2016, Beisser et al., 2010], the approach proposed in this paper also provides a dual bound which allows to judge the quality of the attained (primal) solutions; furthermore, the proposed scheme can be embedded within a branch-and-bound framework, allowing an exact resolution of the problem. Note that implementations of Lagrangian relaxation-based algorithms have been successfully applied to solve problems related to the MWCS. Sophisticated Lagrangian relaxation schemes (without cut generation) are designed in [Haouari et al., 2008, 2010] for the PCST. Complementary, R&C implementations are devised in [Lucena, 2005, 2006] for the Steiner tree problem, and in [da Cunha et al., 2009] for the PCSTP.

In order to assess the efficiency of the R&C algorithm proposed in this paper, both from the view of solution quality and runtime, a computational study on a large set of benchmark instances from the literature is reported. Moreover, additional large-scale instances, which have been constructed to resemble interactomes, are tested as well. The implemented program provided for download at Sinnl and Álvarez-Miranda.

Paper Outline In Section 2 the ILP formulation used in the R&C algorithm is presented; likewise, the cardinality- and budget-constrained versions are discussed in more detail. A generic scheme of R&C is outlined in Section 3. In Section 4 the designed algorithmic framework is described. Computational results are reported in Section 5. Finally, concluding remarks are drawn in Section 6.

2 An ILP Formulation for (Variants of) the MWCS

In its simplest form, the MWCS can be defined as follows.

Definition 1. ***(The MWCS)*** *Given an undirected graph $G = (V, E)$, with weight function $\mathbf{w} : V \to \mathbb{R}$, the MWCS is the problem of finding a subgraph $T = (V_T, E_T)$ that maximizes the total weight $\sum_{v \in V_T} w_v$, and every pair $\{i, j\} \in V_T$ can be connected by a path composed exclusively by edges contained in E_T.*

As mentioned before, this definition can be complemented by so-called *side* constraints depending on the particular application. For instance, in some contexts, besides the weight function one has to additionally consider a cost or penalty function $\mathbf{c} : V \to \mathbb{N}$ and a budget bound $B \in \mathbb{N}$. Hence, the sought maximum weight connected subgraph must also respect a budget constraint $\sum_{v \in V_T} c_v \leq B$. Such constraint may appear, for instance, in Bioinformatic settings where *compact*, i.e., cardinality constrained, functional modules are preferred over large ones [see, e.g. Yamamoto et al., 2009, Yosef et al., 2011]; in this case $\mathbf{c} = \mathbf{1}$. Likewise, in a wildlife corridor design setting although the aim is to find a connected reserve that maximizes the ecological suitability, it must respect an economical bound [see, e.g. Dilkina and Gomes, 2010]. Similarly, in the design of wireless networks, although the objective is to construct a mesh that maximizes the service coverage, there are construction budgets that must be satisfied [see, e.g. Kuo et al., 2015]. Although the cardinality-constrained version is a special case of the budget-constrained version, here it is regarded as its own problem variant, as it allows to use a more efficient solution approach. In the budget-constrained version, one assumes that $c_i \leq B$, $\forall i \in V$, as nodes not fulfilling this can easily be removed at the beginning.

Let $\mathbf{y} \in \{0, 1\}^{|V|}$ be a binary variable such that $y_i = 1$ if node $i \in V$ is part of the connected subgraph, and $y_i = 0$ otherwise. Let Φ denote the set of all $\{0, 1\}^{|V|}$ vectors associated with connected components of G. With this notation, a general version of the MWCS can be modeled as

$$\max \left\{ \mathbf{w}^T \mathbf{y} \mid \Lambda \mathbf{y} \leq \boldsymbol{\beta} \, , \mathbf{y} \in \Phi \text{ and } \mathbf{y} \in \{0, 1\}^{|V|} \right\}, \tag{1}$$

where $\Lambda \mathbf{y} \leq \boldsymbol{\beta}$ represents a (possibly empty) set of side constraints.

There are several alternatives to model the constraint $\mathbf{y} \in \Phi$; in this paper a node-based model, as proposed in [Álvarez-Miranda et al., 2013a, El-Kebir and Klau, 2014, Fischetti et al., 2014], is considered. For formulating such model, the following definition is needed.

Definition 2. ***(Node-separator)*** *For two distinct nodes k and ℓ from V, a subset of nodes $N \subseteq V \setminus \{k, \ell\}$ is called* (k, ℓ)-node separator *if and only if after eliminating N from V there is no (k, ℓ) path in G. A separator N is* minimal *if $N \setminus \{i\}$ is not a (k, ℓ) separator, for any $i \in N$. Let $\mathcal{N}(k, \ell)$ denote the family of all minimal (k, ℓ) separators.*

Obviously, if $\exists \{k, \ell\} \in E$ or if ℓ is not reachable from k, it holds that $\mathcal{N}(k, \ell) = \emptyset$. Using Definition 2 and the previous notation, the connectivity requirement represented by $\mathbf{y} \in \Phi$ can be encoded by the following set of constraints

$$\sum_{j \in N} y_j \geq y_k + y_\ell - 1, \ \forall N \in \mathcal{N}(k, \ell), \ \forall k, \ell \in V \tag{CONN.1}$$

$$\mathbf{y} \in \{0, 1\}^{|V|}. \tag{CONN.2}$$

Constraints (CONN.1) account for the connectivity of the subgraph induced by those variables verifying $y_\ell = 1$ and $y_k = 1$: if t and ℓ are part of the solution, then at least one of the nodes from N has to be taken. The nature of the variables is imposed by (CONN.2). Note that constraints (CONN.1) are exponential in number. Hence, in the proposed algorithm, only few constraints (CONN.1) are added at the beginning; the remaining constraints are *separated and dualized on-the-fly*. This approach is known as Relax-and-Cut [Lucena, 2005], which is outlined in the next section.

Compared to the formulation proposed by [Álvarez-Miranda et al., 2013a] for the MWCS, the model based on (CONN.1) does not make use of an artificial root node for modelling connectivity.

Finally, this section is complemented with the following helpful observation [Fischetti et al., 2014].

Observation 1. *Let $P := \{i \in V \mid w_i > 0\}$ be the nodes with positive weights. There is always an optimal solution to the MWCS and its cardinality- or budget-constrained version, where only nodes in P have degree one.*

Proof. Suppose a subgraph T', which contains a node $i \in V \setminus P$ with degree one, is the optimal solution. Since i has degree one (and $c_i \geq 0$), it can be removed, and the solution stays feasible. Since $w_i < 0$, the objective of the solution with i removed is better than the objective of T', which is a contradiction to the optimality of T'. □

Based on this observation, the constraints

$$\sum_{(i,k)\in E} y_i \geq 2y_k, \ \forall k \in V \setminus P, \tag{NON-LEAF}$$

are valid inequalities for the problem.

3 Relax-and-Cut Optimization Scheme

Let

$$\text{CO}: \quad z^* = \max\left\{\mathbf{c}^T\mathbf{y} \mid A\mathbf{y} \leq \mathbf{b}, \ D\mathbf{y} \leq \mathbf{e} \text{ and } \mathbf{y} \in \{0,1\}^n\right\}, \tag{CO}$$

be a generic combinatorial optimization problem of n binary variables. Assume that $A\mathbf{y} \leq \mathbf{b}$ defines a system of so-called *complicating* constraints, i.e., the problem without these constraints is easy to solve. In such case, a classical approach in optimization is to relax $A\mathbf{y} \leq \mathbf{b}$ in a Lagrangian way. Let $\boldsymbol{\lambda} \in \mathbb{R}^m_{\geq 0}$ be a vector of multipliers associated with the m constraints encoded by $A\mathbf{y} \leq \mathbf{b}$. For a given vector $\boldsymbol{\lambda}$, the Lagrangian relaxation of (CO) is given by

$$z_R(\boldsymbol{\lambda}) = \max\left\{\left(\mathbf{c}^T + \boldsymbol{\lambda}^T A\right)\mathbf{y} - \boldsymbol{\lambda}^T\mathbf{b} \mid D\mathbf{y} \leq \mathbf{e} \text{ and } \mathbf{y} \in \{0,1\}^n\right\}, \tag{RL}$$

and it verifies $z_R(\boldsymbol{\lambda}) \geq z^*$, i.e., it gives a dual bound. In order to find the best dual bound, one then solves the Lagrangian dual problem

$$z_L^* = \min\left\{z_R(\boldsymbol{\lambda}) \mid \boldsymbol{\lambda} \geq 0\right\} \tag{LD}$$

using, e.g., a subgradient algorithm [see, e.g., Anstreicher and Wolsey, 2007, Kiwiel et al., 2007, Nesterov, 2007, Sherali and Choi, 1996, for discussions regarding subgradient algorithms]. Using such approach, (LD) gets iteratively solved for different $\boldsymbol{\lambda}$; let $\mathbf{y}^t$ be the optimal solution of the t-th iteration.

For some problems, including the one considered in this work, the set $(A, \mathbf{b}) \equiv A\mathbf{y} \leq \mathbf{b}$ is of exponential size, i.e., it is not practical to specify and relax $A\mathbf{y} \leq \mathbf{b}$ all at once. This led to the development of Relax-and-Cut (R&C) algorithms [see, e.g., Engevall et al., 1998, Escudero et al., 1994, Klose, 2000]. A comprehensive presentation of the R&C approach can be found in the seminal works [Aboudi and Jörnsten, 1991, Lucena, 2005, 2006] and [M., 1998].

In R&C, one starts with a (RL) model with a (possibly empty) subset $(\tilde{A}, \tilde{\mathbf{b}})$ of $(A, \mathbf{b})$, and associated multipliers $\tilde{\boldsymbol{\lambda}}$. During the course of optimizing (LD), a set of constraints, say $(A', \mathbf{b}')$, violated by the current $\mathbf{y}^t$, i.e., verifying $A'\mathbf{y}^t > \mathbf{b}'$, is identified by solving a separation problem. Such constraints are then added to $(\tilde{A}, \tilde{\mathbf{b}})$. Note that contrary to an ILP-based branch-and-cut approach, a constraint can be violated in more than one iteration when using a R&C algorithm. In order to add (and dualize) every constraint only once, a *hashset* is used to store all added constraint in order to check if the candidate constraint has been added before.

Let $(A^t, \mathbf{b}^t)$ be the set of separated constraints up to iteration t, and $\boldsymbol{\lambda}^t$ the associated multipliers. Let $\mathbf{w}^t = \mathbf{c} + {\boldsymbol{\lambda}^t}^T A^t$, and observe that the objective of the relaxed problem at iterations t can be written as ${\mathbf{w}^t}^T\mathbf{y} - \text{const}(t)$, where $\text{const}(t) = {\boldsymbol{\lambda}^t}^T\mathbf{b}^t$. An outline of a R&C algorithm based on the subgradient method

Algorithm 1 Basic Relax-and-Cut Algorithm

Input: Problem (CO); initial relaxed constraints $(A^{-1}, \mathbf{b}^{-1})$; maximum number of iterations $t_{\max}$; tolerance ϵ; step-length factor α; step-length update-parameter p.

1: Set $t := 0$, $\boldsymbol{\lambda}^0 := \mathbf{0}$, $\alpha^0 := \alpha$, PB:= $-\infty$, DB:= ∞ and STOP:=FALSE;
2: $\mathbf{w}^0 = \mathbf{c} + {\boldsymbol{\lambda}^0}^T A^{-1}$
3: **repeat**
4: (**Relaxed problem**) Solve the relaxed problem:

$$z^t = \max \left\{ {\mathbf{w}^t}^T \mathbf{y} - \text{const}(t) \mid D\mathbf{y} \leq \mathbf{e} \text{ and } \mathbf{y} \in \{0,1\}^n \right\}. \qquad (\text{RL}^t)$$

Let $\mathbf{y}^t$ be the optimal solution of (RL^t); set DB:= z^t if z^t <DB;
5: (**Separation problem**) Find one (or more) constraints

$$A\mathbf{y}^t \leq \mathbf{b}$$

violated by $\mathbf{y}^t$, let $(A', \mathbf{b}')$ be these constraints. Initialize the corresponding multipliers λ_i^t for $i \in Rows((A', \mathbf{b}') \setminus (A^{t-1}, \mathbf{b}^{t-1}))$. Set $(A^t, \mathbf{b}^t) = (A^{t-1}, \mathbf{b}^{t-1}) \cup (A', \mathbf{b}')$;
6: (**Update PB**) Use $\mathbf{y}^t$ within a Primal Heuristic for finding a primal bound PB$'$; set PB:=PB$'$ if PB$'$ >PB;
7: (**Update DB**) If DB did not change for p iterations; then reset $\mathbf{y}^t$ to the dual incumbent, set $z^t :=$ DB and update α;
8: (**Update subgradient**) Calculate the subgradient g^t associated with $(A^t, \mathbf{b}^t)$, i.e., entries of g^t are given by

$$g_i^t = b_i^t - A_i^t \mathbf{y}^t,\ i \in Rows(A^t, \mathbf{b}^t);$$

9: (**Update multipliers**) Calculate the new Lagrangian multipliers associated with $(A^t, \mathbf{b}^t)$, i.e., $\lambda_i^{t+1} = \max\{0, \lambda_i^t - \theta^t g_i^t\}, i \in Rows(A^t, \mathbf{b}^t)$; where $\theta^t = \alpha \frac{z^t - PB}{\|\mathbf{g}^t\|}$.
10: **if** $\|\mathbf{g}^t\| \leq \epsilon$, or $(\text{DB} - \text{PB}) \leq \epsilon$, or $t = t_{\max}$ **then**
11: set STOP:=TRUE;
12: **else**
13: $\mathbf{w}^{t+1} = \mathbf{c} + {\boldsymbol{\lambda}^{t+1}}^T A^t$, set $t := t + 1$;
14: **until** STOP = TRUE

is given in Algorithm 1. In Algorithm 1, the best primal bound is saved in variable PB and the best dual bound in DB.

A subgradient $\mathbf{g}^t$ of the current iteration t is given by

$$g_i^t = b_i^t - A_i^t \mathbf{y}^t,\ i \in Rows(A^t, \mathbf{b}^t).$$

In the simplest variant of the subgradient method [see, e.g., Wolsey, 1998], this subgradient is used together with a step-length factor $\alpha \in (0, 2]$ to generate the step-length

$$\theta^t = \alpha \frac{z_R\left(\boldsymbol{\lambda}^t\right) - PB}{\|\mathbf{g}^t\|},$$

and the Lagrangian multipliers are updated by the following rule

$$\lambda_i^{t+1} = \max\left\{0, \lambda_i^t - \theta^t g_i^t\right\}.$$

When a constraint (A_i, b_i) gets added to $(A^t, \mathbf{b}^t)$, λ_i^t is initialized equal to 0. Following [Haouari et al., 2008], if there is no improvement of DB for the last $p > 0$ iterations, α gets halved and the current solution $\mathbf{y}^t$ is reset to the dual incumbent solution and $z_R\left(\boldsymbol{\lambda}^t\right)$ to DB, before the calculation of the subgradient.

An alternative method for calculating the multipliers is also considered; the *average direction strategy* (ADS), which is proposed in [Sherali and Ulular, 1990]. This strategy turned out to be the most efficient one in the computational study of different subgradient strategies carried out in [Haouari et al., 2008]. In ADS, a direction $\mathbf{d}^t$ is defined as

$$\mathbf{d}^t = \mathbf{g}^t + \frac{\|\mathbf{g}^t\|}{\|\mathbf{d}^{t-1}\|}\mathbf{d}^{t-1},$$

and the multipliers are be updated as

$$\lambda_i^{t+1} = \max\left\{0, \lambda_i^t - \theta^t d_i^t\right\}.$$

In this case, the directions d_i^{t-1} are initialized to g_i^{t-1} when a constraint (A_i, b_i) gets added.

4 A Relax-and-Cut Framework for (Variants of) the MWCS

In order, to successfully apply the R&C approach to the MWCS (and its variants), the following steps of Algorithm 1 need to be discussed: (i) solving the relaxed problem (line 4); (ii) finding inequalities violated by the solution of the relaxed problem (line 5); and (iii) trying to obtain a better primal solution (line 6). Procedure (ii) is first described since it is the same for all three variants of the MWCS considered in this paper; afterwards, and for each problem variant, (i) and (iii) are presented. Finally, additional enhancements of the framework are discussed.

4.1 Solving the Separation Problem

Let G^t be the subgraph induced by $V^t = \{i \in V \mid y_i^t = 1\}$, and let $H_1^t, H_2^t, \ldots, H_l^t$ be the l connected components contained in G^t (if there is only one component, then no inequality (CONN.1) is violated by the current solution). For a given component H_i^t, let $\bar{H}_i^t$ be the set of neighboring nodes of H_i^t in G, i.e., $\bar{H}_i^t = \{i' \in V \setminus H_i^t \mid \exists\{i', j\} \in E, j \in H_i^t\}$. A minimal separator between a node $k \in H_k^t$ and a node $\ell \in H_\ell^t$ can be found as follows: (i) delete from G all arcs induced by $H_k^t \cup \bar{H}_k^t$ and $H_\ell^t \cup \bar{H}_\ell^t$; (ii) apply a breadth-first search from k, and let $R(k)$ be the set of all the reached nodes; finally, (iii) the set $\mathcal{N}_{k,\ell} = R(k) \cap \bar{H}_\ell^t$ defines a minimal (k, ℓ)-node separator.

In the case of large-scale instances, searching minimal separators can be computationally costly. As an alternative, one can simple use the neighboring node sets $\bar{H}_u^k$ as node separators. Although not minimal, these separators can be computed very fast. Moreover, the following lifted version of (CONN.1) can be used:

Theorem 1. *Let z^I be the value of a feasible solution, $z^H = \sum_{i \in H} \max(0, w_i)$ for $H \subset V$ and $\bar{H} = \{i' \in V \setminus H | \exists\{i', j\} \in E, j \in H\}$. If $z^H < z^I$, the following inequalities do not exclude the optimal solution of the MWCS (nor of its cardinality- and budget-constrained variants)*

$$\sum_{j \in \bar{H}} y_j \geq y_k, \ \forall k \in H. \tag{L-CONN}$$

Proof. It holds that $z^H < z^*$, which means that there is not enough weight associated with nodes in H so that the optimal solution can be contained in H alone. This means that at least one node ℓ outside of H must be in the solution. If a node $k \in H$ is in the solution, it must be connected to ℓ, and the connection to any node outside of H must use a node in $\bar{H}$. □

4.2 Solving the Relaxed Problems and Primal Heuristics

MWCS In the case of the simple MWCS, the relaxed problem in the t-th iteration is given by

$$z^t = \max\left\{\mathbf{w}^{t^T}\mathbf{y} + \text{const}(t) \mid \mathbf{y} \in \{0,1\}^{|V|}\right\},$$

therefore, its resolution is trivial. Simply pick all nodes such that the corresponding values in $\mathbf{w}^t$ are positive. Note that *picking* a node $i \in V$ means setting $y_i = 1$.

To construct primal solutions based on $\mathbf{w}^t$, a variant of the PrimI-heuristic [de Aragão and Werneck, 2002], originally proposed for the Steiner tree problem, is developed. Recall that in the Steiner tree problem, one is given a graph with positive edge costs and terminal nodes $T \subset V$. The goal is to find a minimum cost subtree containing all nodes in T and potentially some other nodes $V \setminus T$. The heuristic PrimI is a combination of Prim's algorithm for the spanning tree problem and Dijkstra's algorithm for the shortest path problem: a partial solution (i.e., a subtree) G_H is kept throughout the algorithm. In the beginning G_H is initialized with a node $k \in T$; then, shortest paths from G_H to all terminals $k' \in T, k' \neq k$ are computed. Let k^* be the terminal node with cheapest connection cost to G_H. The node k^* and all nodes on the shortest path to it are added to G_H. This procedure is repeated until all nodes in T are connected. Note that the algorithm also works in the directed case.

The heuristic was adapted as follows: the set T is defined as all nodes i with $y_i^t = 1$. The cost of an edge $e : \{i, j\} \in E$ is defined as 0 if $y_j^t = 1$ and $\min\{0, -w_j^t\}$, otherwise. PrimI is ran on this graph to find a Steiner tree; the nodes in the tree are the heuristic solution for the MWCS. As root node to initialize G_H, a node i with maximal w_i^t amongst all nodes in P within the component of G^t, which contains the maximum weight, is chosen. A post-processing procedure was also implemented: note that whenever a node $k \in T$ is added to the partial solution G_H, the solution value z^H may increase or decrease, depending on the weight w_i of the nodes i on the path to k, i.e., it could be better to not connect some nodes $k \in T$. Thus, one must keep track of the value z^H throughout the run of the heuristic, and the partial solution with the highest z^H is taken. Finally, and as an additional post-processing, one can check if there is some node $i \in P\,|_{w_i^t<0}$ adjacent to the obtained solution (recall that such a node is not taken in T); if yes, one can add such node, and repeat the process recursively.

Cardinality-constrained MWCS In the case of the side constraints defining a cardinality requirement $\mathbf{1}^T\mathbf{y} \leq \beta$, the relaxed problem

$$z^t = \max\left\{\mathbf{w}^{t^T}\mathbf{y} + \text{const}(t) \mid \mathbf{1}^T\mathbf{y} \leq \beta \text{ and } \mathbf{y} \in \{0,1\}^{|V|}\right\},$$

can be solved straightforwardly: sort nodes decreasingly by w_i^t and pick nodes until either $w_i^t \leq 0$ or β nodes have been picked.

The primal heuristic is similar to the one for the simple MWCS. The cardinality requirement is taken into account when a node in T is candidate to be added to G_H; if adding it would exceed the cardinality bound, it does not get added.

Budget-constrained MWCS If the side constraints correspond to a single budget constraint $\boldsymbol{c}^T\mathbf{y} \leq B$, the relaxed problem corresponds the following knapsack problem

$$z^t = \max\left\{\mathbf{w}^{t^T}\mathbf{y} + \text{const}(t) \mid \boldsymbol{c}^T\mathbf{y} \leq B \text{ and } \mathbf{y} \in \{0,1\}^{|V|}\right\};$$

this problem is solved by the well-known dynamic programming algorithm [see, e.g., Kellerer et al., 2004, Martello et al., 1999] (after removing nodes i with $w_i^t < 0$).

The primal heuristic is again similar to the one for the simple MWCS. The cost of an edge $e : \{i, j\} \in E$ is set to 0 for all nodes with $y_j^t = 1$, and to c_j for all nodes with $y_j^t = 0$. If adding a node in T to G_H would exceed the budget B, it does not get added.

4.3 Algorithmic Enhancements

In order to improve the performance of the described R&C algorithm, the following enhancements are incorporated.

Variable Fixing A variable fixing procedure based on Lagrangian multipliers [see, e.g. Wolsey, 1998] is used to reduce the size of the instance throughout the course of the algorithm. Let PB be the best primal

bound, $\mathbf{y}^t$ be the current dual solution, let $\mathbf{w}^t$ be the current vector of reduced costs, and let z^t be the optimal solution value obtained with $\mathbf{w}^t$. For every node i with $y_i^t = 1$, one can set $y_i = 0$ and resolve the problem using $\mathbf{w}^t$, if the value of the solution does not exceed PB, y_i needs to be 1 in every optimal solution and one can fix $y_i = 1$. The same arguments also hold for $y_i^t = 0$, setting it to $y_i = 1$ and resolving.

Resolving the MWCS and its cardinality-constrained variant can be done very efficiently, as it does not need to be done from scratch for each node. However, for the budget-constrained MWCS, the Dantzig-bound is calculated instead of resolving the problem exactly. The bound is obtained by ordering the elements by w_i/c_i in descending order and picking items until the budget B is exhausted (note that the last item may be picked only fractionally) [see Martello and Toth, 1990, for further details]. As this gives an upper bound, the arguments for variable fixing are still valid. The resolving procedures are as follows.

- MWCS: as there are no side constraints, setting $y_i = 0$ for a node with $y_i^t = 1$ leads to $\mathbf{y}^t$ with $y_i^t = 0$ as optimal solution of the resolved problem. The objective of this solution is $z^t - w_i^t$ (which verifies $z^t - w_i^t < z^t$, since $w_i^t > 0$ due to $y_i^t = 1$). Setting $y_i = 1$ for a node with $y_i^t = 0$ follows similar arguments.

- Cardinality-constrained MWCS: the optimal solution needs to respect the cardinality constraint. Thus, when $y_i = 0$ is set for a node with $y_i^t = 1$, an additional node can be added to the solution. Let j' be the node with largest w_j^t amongst all nodes $j \in V$ with $y_j^t = 0$. Clearly, the optimal solution is obtained by adding j' to $\mathbf{y}^t$ with $y_i^t = 0$, if $w_{j'}^t > 0$ and doing nothing otherwise. The objective of this solution is $z^t - w_i^t + \max\{0, w_{j'}^t\}$. When a node with $y_i^t = 0$ is set to $y_i = 1$, a node with $y_i^t = 1$ needs to be zero due to the cardinality requirement (if the corresponding constraint is tight). Clearly, to obtain the optimal solution, the node j' with minimal $w_{j'}^t$ amongst all nodes with $y_i^t = 1$ needs to be set to zero. This leads to the objective value of $z^t + w_i^t - w_{j'}^t$. Note that $w_i^t \leq w_{j'}^t$,thus the value is $\leq z^t$. If the constraint that defines the cardinality is not tight (this implies $w_i^t < 0$), one would obtain $z^t + w_i^t$.

- Budget-constrained MWCS: the optimal solution needs to respect the budget requirement. Setting $y_i = 0$ for a node with $y_i^t = 1$ can be incorporated in the calculation of the Dantzig-bound by skipping the node. Setting $y_i = 1$ for a node with $y_i^t = 0$ can be dealt with by reducing the budget B by c_i and adding w_i to the obtained Dantzig-bound.

Cut Management For a set of nodes V^t, not all the constraints (CONN.1) between each pair of components H_i^t, H_j^t, are added to the model. This is done since preliminary experiments showed bad performance when adding too many cuts; a similar behavior has been observed in [da Cunha et al., 2009], where subtour-elimination constraints are used in a R&C framework. Instead, the components are sorted in decreasing way according to the sum of positive weights of the nodes comprising them, and a constraint (CONN.1) is added only between two consecutive components, say H_k^t and H_ℓ^t, in this sorted list. A node k (ℓ) with maximal w_k^t (w_ℓ^t) amongst all nodes in $H_k^t \cap P$ ($H_\ell^t \cap P$) is taken as argument for the right-hand-side of an added cut in case of the classic MWCS and the cardinality-constrained MWCS. For the budget-constrained MWCS, the ratio w_k^t/c_k (w_ℓ^t/c_ℓ) is taken as criterion.

The *aging procedure* suggested by [da Cunha et al., 2009] is also incorporated into the proposed algorithmic scheme. The procedure is based on the observation that good practical convergence of a subgradient approach (in a classical, i.e., not R&C setting) can be achieved by setting $g_i^t = 0$, when $\lambda_i^t = 0$ [see Beasley, 1993]. The idea is that a constraint i with $g_i^t > 0$ and $\lambda_i^t = 0$ only influences the step-size α^t, but not the Lagrangian cost $\mathbf{w}^t$, which can lead to a bad performance. Following the suggested aging procedure, g_i^t is set to 0 if in the previous τ iterations, the associated Lagrange multiplier has been zero, i.e, $\lambda_i^{t-\tau} = \ldots = \lambda_i^t = 0$. Additionally, d^{t-1} is also set to 0, since the step-size is also influenced by this value in the considered ADS.

Finally, the inequalities (NON-LEAF) and (L-CONN), for $H = i \in P$, are dualized at the start of the algorithm. The latter constraints exclude single-node solutions being feasible; however, the best single-node solution can be easily found at the start of the algorithm, and PB is initially set to its weight.

Preprocessing The following simple preprocessing procedures have been implemented;

- Removal of all zero-degree nodes.
- Removal of all connected components H, where $\sum_{i \in H} \max\{0, w_i\} \leq w^*$ and $w^* = \max_{i \in V} w_i$.
- Recursive removal of all nodes in $V \setminus P$ with degree one (following Observation 1).

5 Computational Results

The R&C framework was implemented in C++. The program is available online at [Sinnl and Álvarez-Miranda]. Runs were carried out on an Intel Xeon CPU with 2.5 GHz and 12GB memory. The step-length factor α is initially set to 2, and halve it after $p = 20$ iterations without improvement. As parameter τ in the aging procedure, the value three is used. The heuristics are called at every iteration and the variable fixing routines are called whenever the primal or dual bound improves.

5.1 Benchmark Instances

MWCS For the MWCS, the following three recently published datasets, which can be downloaded from [11th DIMACS Implementation Challenge, 2014], are used:

- `ACTMOD` This is a group of 8 instances from integrative biological network analysis, in which the goal is to find active modules. They were originally proposed by [El-Kebir, 2015, El-Kebir and Klau, 2014]. The size of the instances ranges from 2034 nodes and 7756 edges, to 5226 nodes and 93394 edges.
- `Hand` This is a group of 48 instances created from images of hand-written text and motivated by an application of the PCSTP problem in signal processing. These instances were introduced in [Hegde et al.]. The size of the instances ranges from 39600 nodes and 78704 edges, to 169800 nodes and 338551 edges. Note that these instances are originally for the PCSTP; however, since they have uniform edge weights, they can be easily transformed into MWCS-instances.
- `JMPALMK` This is a group of 72 random Euclidean instances with a topology similar to street networks and they were first presented in [Álvarez-Miranda et al., 2013a]. They are generated as proposed in [Johnson et al., 2000]. First, n nodes are randomly located in a unit Euclidean square. A link between two nodes i and j is established if the Euclidean distance d_{ij} between them is no more than $\rho/\sqrt{n}$, for a fixed $\rho > 0$. For a given n and a given ρ, node weights are generated according to the following procedure: $\delta\%$ of the nodes are randomly selected to be associated with non-zero weights; out of them, an $\epsilon\%$ of nodes is associated with a weight taken uniformly randomly from $[-10, 0]$, and the remaining nodes are associated with a weight taken uniformly randomly from $[0, 10]$ ($\delta, \epsilon \in \{0.25, 0.50, 0.75\}$) The size of the instances ranges from 500 nodes and 2597 edges, to 1500 nodes and 20527 edges.

Besides the above described instance datasets, an additional set of synthetic large-scale instances was also used in the computations. These instances have been created to resemble gene interaction networks (i.e., *interactomes*). It is well-known that interactomes verify specially structured topologies such as scale-free, geometric and Erdös-Rényi random graphs [see, e.g. Albert, 2005, Friedel and Zimmer, 2006, Janjić et al., 2008, Przulj et al., 2004, Rajagopalan and Agarwal, 2005]. The graphs have been created with functions available in the R package `igraph` [Csardi and Nepusz, 2006]. For each type of graph, instances are generated setting $|V|$ equal to 10000 and 20000, and using three values of edge density. This leads to instances with 10000 nodes and 5246096 edges, up to 20000 nodes and 20951806 edges.

For each of the resulting networks, node scores are generated following the ideas presented in [Dittrich et al., 2008, Rajagopalan and Agarwal, 2005]; namely,

1. An artificial subnetwork is identified by performing a random walk of a length chosen uniformly random in $(100, 200)$.

2. Nodes in the encountered subnetwork receive a p-value[1] chosen uniformly random in $(0, 10^{-3})$, all other nodes get a p-value chosen uniformly random in $(0, 1)$.

3. The scoring function proposed by [Dittrich et al., 2008], and available in `BioNet` [see Beisser et al., 2010], is used to generate scores using the assigned p-values and a specified desired false-discovery-rate of 0.2.

These instances are made available online at [Sinnl and Álvarez-Miranda] and will be denoted as `synthetic-interactomes`.

Cardinality-constrained MWCS The instances from the sets `ACTMOD` and `synthetic-interactomes` are used to test the algorithm for the cardinality-constrained MWCS. For each instance, the cardinality bound is set to $10\%, 25\%, 50\%$ of the number of nodes with positive weight, i.e., $\beta = \{0.1|P|, 0.25|P|, 0.5|P|\}$.

Budget-constrained MWCS For the budget-constrained MWCS, the set `Wildlife`, originally proposed in [Dilkina and Gomes, 2010], is considered. This is a group of random grid graph-based instances, which aim at resembling wildlife habitat zones where maximum suitability corridors must be designed. Each instance is given by an $L \times L$ grid, and weights and costs are randomly taken from $\{0, \dots, 10\}$. Depending on the relation between weight and cost, instances are classified as `weak` (weakly correlated) and `uncor` (uncorrelated). The largest instances from this set have been taken as part of the computations; L is set to 20, which leads to networks comprised by 400 nodes and 760 edges. These instances originally have terminals that must be in any feasible solutions. This has been ignored in order to transform the instances to be suitable for the budget-constrained MWCS considered in this paper.

Complementary, a set of large-scale instances, denoted as `synthetic-budget`, has been also generated. Similar to the set `synthetic-interactome`, networks corresponding to scale-free, geometric and Erdös-Rényi random graphs were created using the R package `igraph`. For each type of graph, instances were generated setting $|V|$ equal to 1000, 5000 and 10000, and using three values of edge density. Weights are randomly taken from $\{-10, \dots, 10\}$ and costs from $\{0, \dots, 10\}$. For each instance, computations were carried out considering $B = \{0.05C_{total}, 0.10C_{total}, 0.20C_{total}\}$, where $C_{total} = \sum_{i \in V} c_i$.

5.2 Results for MWCS

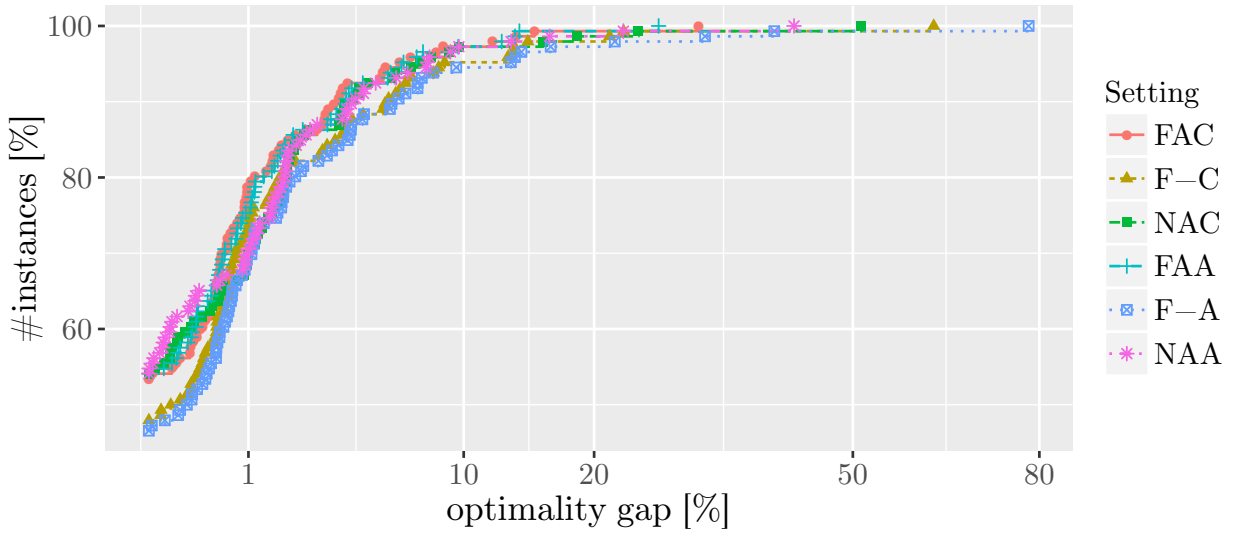
The effect of different ingredients of the proposed R&C approach are first compared; the result of such comparison is reported in Figure 1, which gives plots of the optimality gap for six different settings obtained by combining these ingredients when solving the instances from sets `ACTMOD`, `JMPALMK`, `synthetic-instances` and `hand`. A given setting is labeled as XYZ: X corresponds to F if the *fast separation* (together with Theorem 1) is used, or to N if the normal separation of connectivity cuts (CONN.1) is used; Y corresponds to A if the *aging procedure* for cuts is used, or "−" if it is not used; and Z indicates whether the classic subgradient (denoted by C) or the average direction strategy ADS (denoted by A) is used. In this plot, the optimality gap is calculated as $100 \cdot (\text{UB} - z^*)/z^*$, where UB is the obtained upper bound and z^* is the value of the best solution found by the setting. As the instance set `hand` is originally of a minimization form (since it is a uniform PCSTP), the gap for these instances is calculated as $100 \cdot (z^* - \text{LB})/z^*$, where LB is the obtained lower bound.

The results reported in Figure 1 show that using the aging procedure seems to have a positive influence on the gaps. Aside from this, the performance is very similar among the different settings, i.e., the use of fast separation instead of normal separation, or the use of classic subgradient instead of ADS, does not have strong impact (note that the scale in Figure 1 has been transformed by using the square-root for allowing a better readability). Settings FAC and FAA allow to obtain a gap of at most 1% for nearly 80% of the MWCS instances. Based on the figure, the setting FAC is used for all further experiment.

Next, the performance of the proposed R&C algorithm is compared against a state-of-the-art ILP-based framework of [Fischetti et al., 2014] The framework proposed by [Fischetti et al., 2014] was winner in many categories of the recent 11th DIMACS Implementation Challenge on Steiner trees, including the MWCS

[1] a smaller p-value indicates a higher confidence that the gene is part of the searched subnetwork

Figure 1: Optimality gaps obtained by different settings of the R&C approach.



category [11th DIMACS Implementation Challenge, 2014]. Note that the runs in [Fischetti et al., 2014] were carried out on the same machine as the runs in this work, with a timelimit of 3600 seconds and multi-threading using four threads. Since the framework is available, results on the `synthetic-interactomes` are also reported in this paper.

In the following Tables, results for the MWCS are shown. In particular, the upper bound UB (resp. lower bound LB for `hand`), the value of the best found solution z^*, the optimality gap g[%], the runtime in seconds $t[s]$ and the primal gap pg[%] of the best solution found by the proposed R&C against the best solution reported in [Fischetti et al., 2014], are reported when solving the MWCS on the different datasets. The primal gap is calculated as

$$100 \cdot (z^{\text{Fischetti et al. [2014]}} - z^*)/z^{\text{Fischetti et al. [2014]}},$$

for `ACTMOD`, `JMPALMK`, and `synthetic-instances`, and as

$$100 \cdot (z^* - z^{\text{Fischetti et al. [2014]}})/z^{\text{Fischetti et al. [2014]}},$$

for `hand`; $z^{\text{Fischetti et al. [2014]}}$ corresponds to the value of the best solution found by the approach proposed in [Fischetti et al., 2014]. Note that a negative value for pg[%] means the primal solution obtained by the R&C scheme improves the solution found by [Fischetti et al., 2014]. This happens for instances of the set `hand`, for which the approach proposed in [Fischetti et al., 2014] was not able to solve these instances to optimality. Additionally, observe that the approach designed in [Fischetti et al., 2014] crashes for some instances of the set `synthetic-interactomes` due to memory problems.

Table 1 shows the results for instance set `ACTMOD`. For one instance, *lymphoma*, the proposed R&C approach proves optimality, and for three additional instances it finds the optimal solution (*HCMV*, *metabol_expr_mice_2*, and *metabol_expr_mice_3*). For three additional instances, the primal gap is under 2% (*drosophila005*, *drosophila0075*, and *metabol_expr_mice_1*). Instance *drosophila001* causes the most trouble with a primal gap of 8.76% and an optimality gap of 30.40%. In all cases, the runtime is at most 3 seconds, while the longest runtime reported by [Fischetti et al., 2014] is eleven seconds (this occurs for *drosophila005*).

The results obtained when solving the MWCS on the instance set `Hand` are provided in Table 2. The largest optimality gap is 8.05% and it is attained for instance *handsd02*; however, the optimality gap is under 1% for most instances. Optimality is certified by the R&C for four instances (*handsd10, handbd14, handbi12*

Table 1: Results for instance set ACTMOD. z^* gives the value of the best solution found, UB the obtained upper bound, $t[s]$ the runtime (in seconds), and $g[\%]$ the optimality gap; $z^{\text{Fischetti et al. [2014]}}$ is the value of the best solution reported in [Fischetti et al., 2014], $t_{\text{Fischetti et al. [2014]}}[s]$ the runtime (in seconds) reported in [Fischetti et al., 2014] (TL indicates that the timelimit of 3600 seconds had been reached), and $pg[\%]$ is the primal gap between the best solution value found by the proposed R&C and the one in [Fischetti et al., 2014].

instance	$\|V\|$	$\|E\|$	z^*	UB	$t[s]$	$g[\%]$	$z^{\text{Fischetti et al. [2014]}}$	$t_{\text{Fischetti et al. [2014]}}[s]$	$pg[\%]$
drosophila001	5226	93394	22.422342	29.258480	2	30.49	24.385518	10	8.76
drosophila005	5226	93394	177.705532	182.791076	3	2.86	178.664008	11	0.54
drosophila0075	5226	93394	256.401506	264.277715	2	3.07	260.523591	8	1.61
HCMV	3863	29293	7.554315	8.687654	1	15.00	7.554315	1	0.00
lymphoma	2034	7756	70.166309	70.166309	0	0.00	70.166309	0	0.00
metabol_expr_mice_1	3523	4345	540.093679	613.606515	1	13.61	544.948342	2	0.90
metabol_expr_mice_2	3514	4332	241.077524	269.814979	0	11.92	241.077525	1	0.00
metabol_expr_mice_3	2853	3335	508.260877	550.629656	0	8.34	508.260879	1	0.00

and *handbi14*); additionally, the optimal solution is found for 21 out the 44 instances. Moreover, for 12 instances, where the algorithm of [Fischetti et al., 2014] does not find the optimal solution, the proposed R&C allows to compute an improved solution. The longest runtime of the R&C approach is 50 seconds and occurs for *handbd13*. Moreover, for the instances in which the R&C is able to prove optimality, it does it usually much faster than the approach devised by [Fischetti et al., 2014]; the most extreme case being *handbi12* with 6 seconds against 3379 seconds.

Tables **??** and **??** (in the Appendix **??**) show the results for the JMPALMK instances. As these instances turned out to be rather easy for the proposed approach (it is possible prove optimality for nearly all of them in less than a second), the results obtained by Fischetti et al. [2014] (which has also been able to prove optimality for all of these instances in a few seconds) are not reported. Therefore, these instances seem to be easy for both approaches.

Finally, results for set synthetic-instances are given in Table 3. The designed R&C is capable of providing guarantee of optimality for 15 out of 18 instances, and for the remaining three, optimal solution is found. The runtime for 11 instances is at most one second, and only for two instances, the R&C needs more than 20 seconds. In contrast to this, the approach given in Fischetti et al. [2014] crashed for seven of the instances due to memory problems caused by the large size of the instances. Moreover, runtimes are up to 107 seconds. These results, along with those for Hand instances, show that the R&C algorithm is capable of tackling massive instances, providing nearly optimal solutions within very short running times for cases where the state-of-the-art exact ILP approach fails in doing so.

Table 2: Results for instance set `Hand`. z^* gives the value of the best solution found, LB the obtained lower bound, $t[s]$ the runtime, $g[\%]$ the optimality gap; $z^{\text{Fischetti et al. [2014]}}$ is the value of the best solution reported in [Fischetti et al., 2014], $t_{\text{Fischetti et al. [2014]}}[s]$ the runtime reported in [Fischetti et al., 2014] (TL indicates that the timelimit of 3600 seconds had been reached), and $pg[\%]$ the primal gap between the best solution value found by the proposed R&C and the one in [Fischetti et al., 2014].

instance	$\|V\|$	$\|E\|$	z^*	LB	$t[s]$	$g[\%]$	$z^{\text{Fischetti et al. [2014]}}$	$t_{\text{Fischetti et al. [2014]}}[s]$	$pg[\%]$
handsd01	42500	84475	171.636766	169.855952	8	1.04	171.636766	77	0.00
handsd02	42500	84475	160.200588	147.306408	9	8.05	160.345804	TL	-0.09
handsd03	42500	84475	31.306275	31.208919	6	0.31	31.306275	47	0.00
handsd04	42500	84475	495.373251	463.364025	8	6.46	494.554042	TL	0.17
handsd05	42500	84475	21.937611	21.858403	7	0.36	21.937611	62	0.00
handsd06	42500	84475	280.623129	275.265819	9	1.91	279.903130	949	0.26
handsd07	42500	84475	11.804120	11.733511	10	0.60	11.804120	68	0.00
handsd08	42500	84475	143.237729	142.956110	9	0.20	143.237729	610	0.00
handsd09	42500	84475	3.818683	3.701843	12	3.06	3.822404	TL	-0.10
handsd10	42500	84475	1034.767360	1034.767360	1	0.00	1034.767359	24	0.00
handsi01	39600	78704	295.453616	293.481793	6	0.67	295.453616	49	0.00
handsi02	39600	78704	125.526771	120.934488	10	3.66	125.429411	2240	0.08
handsi03	39600	78704	56.149422	55.874183	6	0.49	56.149422	57	0.00
handsi04	39600	78704	724.603412	687.440055	9	5.13	722.508202	2863	0.29
handsi05	39600	78704	35.043506	34.912097	6	0.37	35.043506	42	0.00
handsi06	39600	78704	453.072412	448.690460	7	0.97	452.953621	775	0.03
handsi07	39600	78704	18.410135	18.278262	7	0.72	18.410135	56	0.00
handsi08	39600	78704	229.529930	229.354770	8	0.08	229.529930	532	0.00
handsi09	39600	78704	5.962166	5.763122	10	3.34	5.977964	TL	-0.26
handsi10	39600	78704	1803.900570	1800.745620	8	0.17	1803.697508	483	0.01
handbd01	169800	338551	729.024875	716.772497	32	1.68	730.238277	TL	-0.17
handbd02	169800	338551	296.901444	286.243886	38	3.59	297.039586	TL	-0.05
handbd03	169800	338551	135.070605	134.521405	25	0.41	135.070605	555	0.00
handbd04	169800	338551	1839.140710	1742.913550	37	5.23	1820.072929	TL	1.05
handbd05	169800	338551	105.474688	104.819500	28	0.62	105.474688	705	0.00
handbd06	169800	338551	1535.652210	1487.355110	39	3.15	1533.589874	TL	0.13
handbd07	169800	338551	77.861959	76.988786	37	1.12	77.861959	2942	0.00
handbd08	169800	338551	1369.222120	1348.375540	39	1.52	1371.952680	TL	-0.20
handbd09	169800	338551	62.717160	62.111930	30	0.97	62.717160	1004	0.00
handbd10	169800	338551	1137.573620	1131.719420	28	0.51	1139.724449	TL	-0.19
handbd11	169800	338551	46.772533	46.382794	29	0.83	46.772533	720	0.00
handbd12	169800	338551	321.212210	320.664521	36	0.17	321.207482	TL	0.00
handbd13	169800	338551	13.201574	12.695963	50	3.83	13.224294	TL	-0.17
handbd14	169800	338551	4379.104240	4379.104240	1	0.00	4379.104236	45	0.00
handbi01	158400	315808	1358.757560	1335.060210	28	1.74	1360.201440	TL	-0.11
handbi02	158400	315808	532.242362	513.503810	33	3.52	532.616534	TL	-0.07
handbi03	158400	315808	243.134201	242.023536	21	0.46	243.134201	1246	0.00
handbi04	158400	315808	3264.617150	3070.132200	34	5.96	3226.919270	TL	1.17
handbi05	158400	315808	184.467331	183.029955	25	0.78	184.467331	916	0.00
handbi06	158400	315808	2940.136180	2851.396880	28	3.02	2930.642408	TL	0.32
handbi07	158400	315808	150.974258	149.577220	25	0.93	150.974258	1266	0.00
handbi08	158400	315808	2273.079530	2241.922500	26	1.37	2271.575687	TL	0.07
handbi09	158400	315808	107.768806	106.795496	28	0.90	107.768806	986	0.00
handbi10	158400	315808	1874.499740	1870.156450	31	0.23	1874.645163	TL	-0.01
handbi11	158400	315808	68.944709	67.882253	31	1.54	68.953380	TL	-0.01
handbi12	158400	315808	138.257023	138.257023	6	0.00	138.257023	3379	0.00
handbi13	158400	315808	4.357762	4.124417	36	5.35	4.268246	TL	2.10
handbi14	158400	315808	7881.768740	7881.768740	0	0.00	7881.768740	48	0.00

Table 3: Results for instance set `synthetic-instances`, z^* gives the value of the best solution found, UB the obtained upper bound, $t[s]$ the runtime, and $g[\%]$ the optimality gap; $z^{\text{Fischetti et al. [2014]}}$ is the value of the best solution obtained using the framework of Fischetti et al. [2014], $t_{\text{Fischetti et al. [2014]}}[s]$ the runtime, and $pg[\%]$ the primal gap between the best solution value found by the proposed R&C and the one in Fischetti et al. [2014]. For runs, where the framework of [Fischetti et al., 2014] crashed due to needing to much memory, the entries are "-".

instance	$\|V\|$	$\|E\|$	z^*	UB	$t[s]$	$g[\%]$	$z^{\text{Fischetti et al. [2014]}}$	$t_{\text{Fischetti et al. [2014]}}[s]$	$pg[\%]$
ER-10000-1-0-1-0.01-0.2	10000	498886	357.552413	357.552413	0	0.00	357.552413	22	0.00
ER-10000-1-0-1-0.05-0.2	10000	2498648	99.318590	99.318590	1	0.00	-	-	-
ER-10000-1-0-1-0.1-0.2	10000	4998384	105.502362	105.502362	3	0.00	-	-	-
ER-20000-1-0-1-0.01-0.2	20000	1998637	71.001080	71.039088	1	0.05	71.001080	107	0.00
ER-20000-1-0-1-0.05-0.2	20000	9998526	138.724081	138.724081	7	0.00	-	-	-
ER-20000-1-0-1-0.1-0.2	20000	20002644	47.260212	47.260212	24	0.00	-	-	-
GR-10000-1-0-0-0.05-0.2	10000	376587	154.493407	154.493407	0	0.00	154.493407	20	0.00
GR-10000-1-0-0-0.1-0.2	10000	1438063	203.415185	204.011024	2	0.29	203.415185	98	0.00
GR-10000-1-0-0-0.2-0.2	10000	5246096	242.524905	242.524905	3	0.00	-	-	-
GR-20000-1-0-0-0.05-0.2	20000	1503825	57.145074	57.145074	1	0.00	57.145074	75	0.00
GR-20000-1-0-0-0.1-0.2	20000	5749047	74.014732	74.014732	4	0.00	-	-	-
GR-20000-1-0-0-0.2-0.2	20000	20951806	82.180357	82.180357	25	0.00	-	-	-
SF-10000-1-0-2-15-0.2	10000	149880	344.951567	344.957215	0	0.00	344.951567	7	0.00
SF-10000-1-0-2-20-0.2	10000	199790	212.509892	212.509892	0	0.00	212.509892	10	0.00
SF-10000-1-0-2-25-0.2	10000	249675	269.707319	269.707319	0	0.00	269.707319	12	0.00
SF-20000-1-0-2-15-0.2	20000	299880	147.581101	147.581101	0	0.00	147.581101	15	0.00
SF-20000-1-0-2-20-0.2	20000	399790	19.379227	19.669870	1	1.50	19.379227	21	0.00
SF-20000-1-0-2-25-0.2	20000	499675	90.885148	90.885148	0	0.00	90.885148	25	0.00

Table 4: Results for cardinality-constrained MWCS on instance set ACTMOD. z^* gives the value of the best solution found, UB the obtained upper bound, $t[s]$ the runtime, and $g[\%]$ the optimality gap.

	$\beta = 0.1\|P\|$				$\beta = 0.25\|P\|$				$\beta = 0.5\|P\|$			
instance	z^*	UB	$t[s]$	$g[\%]$	z^*	UB	$t[s]$	$g[\%]$	z^*	UB	$t[s]$	$g[\%]$
drosophila001	11.93586	13.96143	0	16.97	17.41989	24.10594	2	38.38	21.09180	30.95160	1	46.75
drosophila005	47.68835	55.52146	2	16.43	94.95791	106.93578	2	12.61	148.55221	158.68120	2	6.82
drosophila0075	68.86193	77.83925	1	13.04	138.94376	149.68238	2	7.73	216.80481	225.04755	3	3.80
HCMV	5.70381	5.70381	0	0.00	7.25127	8.15419	0	12.45	7.55431	8.67551	0	14.84
lymphoma	19.59748	20.78783	0	6.07	40.80643	43.13054	0	5.70	65.51444	65.89767	0	0.58
metabol_expr_mice_1	297.99577	301.01036	0	1.01	407.13140	466.28499	1	14.53	526.51545	590.45370	0	12.14
metabol_expr_mice_2	156.33362	182.30996	0	16.62	233.63344	258.16655	0	10.50	241.07752	289.86890	0	20.24
metabol_expr_mice_3	230.47309	245.12441	0	6.36	376.93230	414.39087	0	9.94	478.78322	521.99491	0	9.03

5.3 Results for Cardinality-constrained MWCS

In Table 4, results obtained when solving the cardinality-constrained MWCS on instances adapted from the ACTMOD set are adapted. The first interesting observation is that for a fixed value of β, the gaps attained for the different instances notoriously differ among each other. For instance, when $\beta = 0.5|P|$, the algorithm is able to find bounds yielding a 0.58% gap for *lymphoma*, while for instance *drosophila001* the attained gap is 46.75%. This uneven behavior of the algorithm can be explained by the different nature of the biological processes embodied by the different instances. Complementary, when comparing the performance of the algorithm for different cardinality bounds β, one can see that there is a clear worsening of the results as β takes larger values. This second characteristic of the results can be explained by how weights are distributed among the nodes of the corresponding network. It seems that nodes with positive weight are rather far from each other since forcing the network to be larger (by increasing β) translates into a more difficult computational task (due to the need of finding the maximum possible sum of node weights).

In Table 5 the results obtained when solving the cardinality-constrained MWCS on the synthetic-interactomes instances are reported. As for the ACTMOD instances, the performance of the algorithm differs among different instances for a fixed cardinality bound β; moreover, in the case of the synthetic-interactomes instances also the runtimes present considerable differences (see columns $t[s]$). One can observe that the value of β also influences the performance of the algorithm; nonetheless, although the average attained gaps worsen when increasing β from $0.1|P|$ to $0.25|P|$, setting β to $0.5|P|$ seems to reduce the difficulty of the problem. This might be explained by the way that the random walk on the underlying networks is used to assign node weights; it could be the case that symmetries drop when larger cardinalities are sought. On a more global perspective, it is possible to conclude that beyond the differences among instances and among results for different values of β, the R&C approach is able to provide good primal and dual bounds (frequently yielding gaps below 5%) for most of the tested problems.

Table 5: Results for cardinality-constrained MWCS on instance set `synthetic-interactomes`. z^* gives the value of the best solution found, UB the obtained upper bound, $t[s]$ the runtime, and $g[\%]$ the optimality gap.

	$\beta = 0.1\|P\|$				$\beta = 0.25\|P\|$				$\beta = 0.5\|P\|$			
instance	z^*	UB	$t[s]$	$g[\%]$	z^*	UB	$t[s]$	$g[\%]$	z^*	UB	$t[s]$	$g[\%]$
ER-10000-1-0-1-0.01-0.2	64.35716	67.18597	1	4.40	135.62564	143.58949	7	5.87	229.33300	242.89582	5	5.91
ER-10000-1-0-1-0.05-0.2	24.76129	25.25805	1	2.01	47.39819	49.35931	2	4.14	72.49274	75.30466	1	3.88
ER-10000-1-0-1-0.1-0.2	26.67163	29.97935	3	12.40	44.48521	54.62011	1	22.78	79.25772	80.39240	4	1.43
ER-20000-1-0-1-0.01-0.2	15.25576	15.25576	1	0.00	33.25369	33.54380	2	0.87	53.06555	54.68026	2	3.04
ER-20000-1-0-1-0.05-0.2	36.02202	40.73715	24	13.09	67.09434	76.98654	16	14.74	94.87515	111.72053	2	17.76
ER-20000-1-0-1-0.1-0.2	15.51015	15.93467	37	2.74	28.63019	30.39145	53	6.15	35.04539	42.60335	7	21.57
GR-10000-1-0-0-0.05-0.2	34.13088	35.78458	0	4.85	68.55768	70.58212	1	2.95	111.49279	114.21246	0	2.44
GR-10000-1-0-0-0.1-0.2	38.85969	43.30315	3	11.43	85.18690	91.78166	1	7.74	142.71912	144.15512	1	1.01
GR-10000-1-0-0-0.2-0.2	47.76557	50.67346	6	6.09	99.59475	106.23343	11	6.67	162.46019	168.83027	1	3.92
GR-20000-1-0-0-0.05-0.2	16.46276	16.55535	1	0.56	33.11465	35.12204	2	6.06	49.81879	51.46512	1	3.30
GR-20000-1-0-0-0.1-0.2	16.49213	19.21597	7	16.52	36.80858	40.69116	30	10.55	59.75756	61.96024	6	3.69
GR-20000-1-0-0-0.2-0.2	19.39676	21.70977	41	11.92	43.23484	45.56720	40	5.39	66.72102	69.12152	27	3.60
SF-10000-1-0-2-15-0.2	72.17727	75.96669	0	5.25	145.94168	149.86156	1	2.69	235.26933	240.11133	1	2.06
SF-10000-1-0-2-20-0.2	43.11756	44.82079	1	3.95	89.78902	96.11791	4	7.05	150.24015	153.92213	2	2.45
SF-10000-1-0-2-25-0.2	49.42539	50.98842	1	3.16	105.04286	108.68488	2	3.47	179.78835	184.13517	4	2.42
SF-20000-1-0-2-15-0.2	34.30778	34.90345	1	1.74	70.01014	72.63181	1	3.74	108.37597	115.43001	8	6.51
SF-20000-1-0-2-20-0.2	4.10735	4.14278	1	0.86	9.16694	9.16694	1	0.00	13.59771	14.56806	1	7.14
SF-20000-1-0-2-25-0.2	24.22558	24.40853	0	0.76	47.84713	50.59606	1	5.75	72.74342	77.91325	4	7.11

Table 6: Results for budget-constrained MWCS on instance set `synthetic-budget`. z^* gives the value of the best solution found, UB the obtained upper bound, $t[s]$ the runtime, and $g[\%]$ the optimality gap.

	$B = 0.05C$				$B = 0.10C$				$B = 0.20C$			
instance	z^*	UB	$t[s]$	$g[\%]$	z^*	UB	$t[s]$	$g[\%]$	z^*	UB	$t[s]$	$g[\%]$
budgetER-10000-1-0.01	9105	9105	0	0.00	13274	13274	1	0.00	19166	19166	1	0.00
budgetER-10000-1-0.05	8740	8740	1	0.00	12922	12922	1	0.00	18813	18813	1	0.00
budgetER-10000-1-0.1	9008	9008	1	0.00	13110	13110	1	0.00	18998	18998	2	0.00
budgetER-1000-1-0.01	834	857	1	2.76	1285	1285	2	0.00	1876	1877	3	0.05
budgetER-1000-1-0.05	925	926	1	0.11	1374	1374	0	0.00	1975	1975	0	0.00
budgetER-1000-1-0.1	889	889	0	0.00	1301	1301	0	0.00	1899	1899	0	0.00
budgetER-5000-1-0.01	4574	4575	26	0.02	6750	6750	0	0.00	9766	9766	0	0.00
budgetER-5000-1-0.05	4286	4286	0	0.00	6299	6299	0	0.00	9141	9141	1	0.00
budgetER-5000-1-0.1	4369	4369	0	0.00	6406	6406	0	0.00	9336	9336	0	0.00
budgetGR-10000-0-0.05	8817	8817	1	0.00	12947	12947	1	0.00	18769	18769	1	0.00
budgetGR-10000-0-0.1	8817	8817	1	0.00	12947	12947	1	0.00	18769	18769	1	0.00
budgetGR-10000-0-0.2	8817	8817	1	0.00	12947	12947	1	0.00	18769	18769	2	0.00
budgetGR-1000-0-0.05	361	645	2	78.67	720	1084	3	50.56	1354	1713	6	26.51
budgetGR-1000-0-0.1	862	886	1	2.78	1316	1316	2	0.00	1914	1920	3	0.31
budgetGR-1000-0-0.2	904	904	0	0.00	1320	1320	0	0.00	1922	1922	0	0.00
budgetGR-5000-0-0.05	4284	4321	26	0.86	6432	6432	87	0.00	9388	9388	1	0.00
budgetGR-5000-0-0.1	4352	4352	0	0.00	6436	6436	0	0.00	9388	9388	0	0.00
budgetGR-5000-0-0.2	4352	4352	1	0.00	6436	6436	0	0.00	9388	9388	0	0.00
budgetSF-10000-2-15	8593	8599	99	0.07	12750	12751	223	0.01	18639	18639	572	0.00
budgetSF-10000-2-20	8696	8696	45	0.00	12903	12907	186	0.03	18828	18828	424	0.00
budgetSF-10000-2-25	8856	8856	120	0.00	13040	13040	0	0.00	18890	18890	1	0.00
budgetSF-1000-2-15	910	910	0	0.00	1351	1351	0	0.00	1964	1964	0	0.00
budgetSF-1000-2-20	861	861	0	0.00	1268	1268	0	0.00	1822	1822	0	0.00
budgetSF-1000-2-25	864	864	0	0.00	1266	1266	0	0.00	1846	1846	0	0.00
budgetSF-5000-2-15	4385	4386	28	0.02	6493	6493	24	0.00	9424	9424	176	0.00
budgetSF-5000-2-20	4448	4453	25	0.11	6599	6599	53	0.00	9535	9535	0	0.00
budgetSF-5000-2-25	4267	4267	27	0.00	6256	6256	49	0.00	9114	9114	0	0.00

5.4 Results for Budget-constrained MWCS

As previously described, the datasets `synthetic-budget` (similar to `synthetic-interactome`) and `Wildlife` are used to investigate the performance of the R&C to tackle the budget-constrained MWCS.

Table 6 summarizes the results obtained when solving this variant of the MWCS for different budgets B . It is possible to see that, regardless of the value of B, the adaptation of the R&C is capable of computing almost optimal solutions for all instances except of instance *budgetGR-1000-0-0.05*, which seems to be surprisingly difficult. By looking at the attained primal and dual bounds of all instances, one can infer that large gap observed for *budgetGR-1000-0-0.05* is caused by a bad dual bound, which shows that the proposed R&C approach is effective in providing good feasible solutions within very short computing times.

The results obtained when solving the budget-constrained MWCS on the `Wildlife` instances are reported in Tables 7 and 8. It is possible to observe that for both, weakly correlated and uncorrelated instances (Table 7 and 8, respectively), the performance of the algorithm seems to improve as the budget increases. This can be explained by the fact that a larger budget allows to explore an area of the feasible space associated with solutions comprised by more nodes and, therefore, more likely to yield better primal bounds. Additionally, one can conclude that, for these two group of instances (weakly correlated and uncorrelated), the designed algorithm works in an equivalent manner since the attained gaps are similar for a fixed B; moreover, the runtimes are also similar.

In general, the grid topology of the `Wildlife` instances is a key element to explain why these instances are so difficult. Despite the fact that the budget bound B seems to have some influence on how symmetric the solutions in the search space are, the nature itself of the underlying networks seems to be burdensome for the proposed R&C scheme.

Table 7: Results for budget-constrained MWCS on instance set `Wildlife` (Table 1 of 2). z^* gives the value of the best solution found, UB the obtained upper bound, $t[s]$ the runtime, and $g[\%]$ the optimality gap.

	$B = 0.05C$				$B = 0.10C$				$B = 0.20C$			
instance	z^*	UB	$t[s]$	$g[\%]$	z^*	UB	$t[s]$	$g[\%]$	z^*	UB	$t[s]$	$g[\%]$
r-uncor-lat-2f+R-u-20-10-10-3-11	283	351	0	24.03	536	578	0	7.84	889	946	2	6.41
r-uncor-lat-2f+R-u-20-10-10-3-15	278	338	0	21.58	509	587	0	15.32	925	994	1	7.46
r-uncor-lat-2f+R-u-20-10-10-3-16	280	308	0	10.00	501	561	0	11.98	915	968	1	5.79
r-uncor-lat-2f+R-u-20-10-10-3-18	258	297	0	15.12	475	538	0	13.26	868	935	1	7.72
r-uncor-lat-2f+R-u-20-10-10-3-20	290	329	0	13.45	535	575	1	7.48	936	984	1	5.13
r-uncor-lat-2f+R-u-20-10-10-3-21	304	343	1	12.83	549	615	0	12.02	975	1025	1	5.13
r-uncor-lat-2f+R-u-20-10-10-3-27	284	345	0	21.48	534	590	1	10.49	949	979	1	3.16
r-uncor-lat-2f+R-u-20-10-10-3-29	278	322	0	15.83	519	577	0	11.18	884	970	1	9.73
r-uncor-lat-2f+R-u-20-10-10-3-40	289	327	1	13.15	505	561	0	11.09	880	944	1	7.27
r-uncor-lat-2f+R-u-20-10-10-3-43	241	306	1	26.97	463	552	0	19.22	864	946	1	9.49
r-uncor-lat-2f+R-u-20-10-10-3-4	265	310	0	16.98	510	567	0	11.18	920	971	1	5.54
r-uncor-lat-2f+R-u-20-10-10-3-52	281	323	0	14.95	501	582	0	16.17	940	988	1	5.11
r-uncor-lat-2f+R-u-20-10-10-3-53	306	341	1	11.44	520	585	1	12.50	921	985	1	6.95
r-uncor-lat-2f+R-u-20-10-10-3-54	252	311	0	23.41	463	539	1	16.41	847	913	1	7.79
r-uncor-lat-2f+R-u-20-10-10-3-56	297	329	0	10.77	538	595	0	10.59	967	996	0	3.00
r-uncor-lat-2f+R-u-20-10-10-3-67	330	363	0	10.00	565	607	0	7.43	941	1001	1	6.38
r-uncor-lat-2f+R-u-20-10-10-3-69	287	329	0	14.63	510	582	0	14.12	931	993	0	6.66
r-uncor-lat-2f+R-u-20-10-10-3-72	281	317	1	12.81	507	564	0	11.24	907	957	2	5.51
r-uncor-lat-2f+R-u-20-10-10-3-74	285	334	0	17.19	519	576	0	10.98	922	952	2	3.25
r-uncor-lat-2f+R-u-20-10-10-3-76	248	292	0	17.74	456	518	0	13.60	844	915	1	8.41
r-uncor-lat-R-u-20-10-10-3-11	343	377	0	9.91	599	625	0	4.34	963	1003	1	4.15
r-uncor-lat-R-u-20-10-10-3-13	256	302	0	17.97	482	558	0	15.77	898	968	1	7.80
r-uncor-lat-R-u-20-10-10-3-14	254	323	0	27.17	489	575	0	17.59	896	977	2	9.04
r-uncor-lat-R-u-20-10-10-3-19	240	293	0	22.08	457	522	0	14.22	845	913	1	8.05
r-uncor-lat-R-u-20-10-10-3-21	261	320	0	22.61	479	549	0	14.61	888	938	1	5.63
r-uncor-lat-R-u-20-10-10-3-22	315	353	0	12.06	554	594	1	7.22	947	989	1	4.44
r-uncor-lat-R-u-20-10-10-3-27	297	339	0	14.14	567	605	0	6.70	966	1042	1	7.87
r-uncor-lat-R-u-20-10-10-3-3	268	312	1	16.42	487	557	0	14.37	911	957	1	5.05
r-uncor-lat-R-u-20-10-10-3-42	292	371	0	27.05	520	640	1	23.08	961	1054	2	9.68
r-uncor-lat-R-u-20-10-10-3-43	309	337	0	9.06	533	592	1	11.07	940	992	1	5.53
r-uncor-lat-R-u-20-10-10-3-46	274	355	0	29.56	523	619	0	18.36	957	1036	1	8.25
r-uncor-lat-R-u-20-10-10-3-53	278	327	0	17.63	507	580	1	14.40	927	992	2	7.01
r-uncor-lat-R-u-20-10-10-3-56	277	321	0	15.88	510	571	0	11.96	891	946	1	6.17
r-uncor-lat-R-u-20-10-10-3-59	290	338	0	16.55	523	591	0	13.00	946	988	1	4.44
r-uncor-lat-R-u-20-10-10-3-61	330	381	0	15.45	589	639	0	8.49	1009	1046	1	3.67
r-uncor-lat-R-u-20-10-10-3-63	339	357	0	5.31	573	607	0	5.93	989	1001	1	1.21
r-uncor-lat-R-u-20-10-10-3-70	268	322	1	20.15	498	564	0	13.25	890	965	1	8.43
r-uncor-lat-R-u-20-10-10-3-72	289	339	0	17.30	522	592	0	13.41	944	1013	2	7.31
r-uncor-lat-R-u-20-10-10-3-78	259	313	0	20.85	492	555	1	12.80	890	964	1	8.31
r-uncor-lat-R-u-20-10-10-3-7	287	335	0	16.72	506	586	1	15.81	924	1002	1	8.44

Table 8: Results for budget-constrained MWCS on instance set `Wildlife` (Table 2 of 2). z^* gives the value of the best solution found, UB the obtained upper bound,$t[s]$ the runtime and $g[\%]$ the optimality gap

	$B = 0.05C$				$B = 0.10C$				$B = 0.20C$			
instance	z^*	UB	$t[s]$	$g[\%]$	z^*	UB	$t[s]$	$g[\%]$	z^*	UB	$t[s]$	$g[\%]$
r-weak-lat-2f+R-u-20-10-10-3-104	292	324	1	10.96	549	586	1	6.74	992	1043	1	5.14
r-weak-lat-2f+R-u-20-10-10-3-10	286	317	0	10.84	533	583	0	9.38	985	1051	1	6.70
r-weak-lat-2f+R-u-20-10-10-3-18	238	304	1	27.73	454	554	0	22.03	901	997	1	10.65
r-weak-lat-2f+R-u-20-10-10-3-19	243	293	0	20.58	441	525	0	19.05	829	943	0	13.75
r-weak-lat-2f+R-u-20-10-10-3-23	256	318	0	24.22	494	583	0	18.02	951	1047	1	10.09
r-weak-lat-2f+R-u-20-10-10-3-26	285	326	0	14.39	516	594	0	15.12	939	1074	1	14.38
r-weak-lat-2f+R-u-20-10-10-3-30	287	310	0	8.01	522	556	0	6.51	891	977	0	9.65
r-weak-lat-2f+R-u-20-10-10-3-31	253	313	0	23.72	476	570	0	19.75	900	1019	1	13.22
r-weak-lat-2f+R-u-20-10-10-3-33	290	341	0	17.59	542	625	1	15.31	1055	1123	1	6.45
r-weak-lat-2f+R-u-20-10-10-3-34	264	297	0	12.50	472	549	0	16.31	897	974	0	8.58
r-weak-lat-2f+R-u-20-10-10-3-36	259	322	0	24.32	489	585	0	19.63	913	1052	0	15.22
r-weak-lat-2f+R-u-20-10-10-3-38	278	319	0	14.75	524	584	0	11.45	955	1043	1	9.21
r-weak-lat-2f+R-u-20-10-10-3-41	278	330	0	18.71	533	602	0	12.95	995	1071	0	7.64
r-weak-lat-2f+R-u-20-10-10-3-51	275	324	0	17.82	511	585	0	14.48	955	1056	1	10.58
r-weak-lat-2f+R-u-20-10-10-3-55	264	292	0	10.61	480	550	0	14.58	908	1005	1	10.68
r-weak-lat-2f+R-u-20-10-10-3-57	265	299	0	12.83	497	559	1	12.47	970	1030	0	6.19
r-weak-lat-2f+R-u-20-10-10-3-70	270	315	0	16.67	511	573	0	12.13	952	1023	1	7.46
r-weak-lat-2f+R-u-20-10-10-3-73	241	282	0	17.01	458	531	1	15.94	839	961	0	14.54
r-weak-lat-2f+R-u-20-10-10-3-7	293	315	0	7.51	520	574	0	10.38	944	1039	1	10.06
r-weak-lat-2f+R-u-20-10-10-3-81	290	327	0	12.76	519	589	0	13.49	963	1044	0	8.41
r-weak-lat-R-u-20-10-10-3-10	290	339	0	16.90	539	604	0	12.06	1008	1087	1	7.84
r-weak-lat-R-u-20-10-10-3-12	284	315	0	10.92	529	576	0	8.88	957	1022	0	6.79
r-weak-lat-R-u-20-10-10-3-14	258	309	0	19.77	459	553	0	20.48	871	975	1	11.94
r-weak-lat-R-u-20-10-10-3-16	309	343	0	11.00	544	612	0	12.50	1014	1089	1	7.40
r-weak-lat-R-u-20-10-10-3-20	268	316	0	17.91	491	554	1	12.83	893	976	1	9.29
r-weak-lat-R-u-20-10-10-3-23	333	357	0	7.21	580	633	0	9.14	1013	1114	0	9.97
r-weak-lat-R-u-20-10-10-3-24	277	323	0	16.61	526	600	0	14.07	993	1099	0	10.67
r-weak-lat-R-u-20-10-10-3-25	280	299	0	6.79	490	544	0	11.02	921	985	1	6.95
r-weak-lat-R-u-20-10-10-3-26	287	336	0	17.07	555	608	0	9.55	987	1084	1	9.83
r-weak-lat-R-u-20-10-10-3-29	266	305	0	14.66	508	572	0	12.60	966	1064	0	10.14
r-weak-lat-R-u-20-10-10-3-32	291	315	0	8.25	508	571	0	12.40	967	1014	0	4.86
r-weak-lat-R-u-20-10-10-3-36	287	339	0	18.12	509	599	1	17.68	969	1057	0	9.08
r-weak-lat-R-u-20-10-10-3-39	252	312	1	23.81	494	565	0	14.37	925	1018	1	10.05
r-weak-lat-R-u-20-10-10-3-40	257	306	0	19.07	516	563	0	9.11	936	1012	0	8.12
r-weak-lat-R-u-20-10-10-3-45	260	296	0	13.85	469	545	1	16.20	854	961	0	12.53
r-weak-lat-R-u-20-10-10-3-4	264	297	0	12.50	489	542	0	10.84	907	985	1	8.60
r-weak-lat-R-u-20-10-10-3-51	246	303	1	23.17	494	554	0	12.15	915	1009	0	10.27
r-weak-lat-R-u-20-10-10-3-52	260	290	0	11.54	495	556	0	12.32	947	1024	1	8.13
r-weak-lat-R-u-20-10-10-3-57	281	339	0	20.64	533	590	0	10.69	947	1034	1	9.19
r-weak-lat-R-u-20-10-10-3-5	272	325	1	19.49	519	582	1	12.14	949	1038	1	9.38

6 Conclusions and Future Work

In this paper, a unified modeling and algorithmic framework is proposed to solve the maximum weight connected subgraph problem (MWCS) and its cardinality- and budget-constrained variant. The proposed scheme relies on a node-based integer linear programming (ILP) formulation which is tackled by an ad-hoc Lagrangian relaxation combined with constraint generation; a so-called Relax & Cut (R&C) algorithm. The approach is enhanced by additional valid inequalities, lifted valid inequalities using optimality-based arguments, primal heuristics and variable fixing procedures.

Computational results on datasets obtained from the literature and newly generated synthetic large-scale instances show that the designed scheme is able to provide, for the MWCS and the two considered variants, reasonable bounds in short computing times; moreover, the results are quite competitive when compared to those obtained by a state-of-the-art ILP-based approach by [Fischetti et al., 2014]. In particular, it is able to improve the best-known solution value for twelve instances from literature. The considered instances have up to 20000 nodes and up to 20951806 edges.

The proposed framework has the advantage of not requiring the use of any ILP-solver. On the one hand, this feature enables it to quickly calculate bounds for instances whose sizes are burdensome for a state-of-the-art ILP-based approaches. On the other hand, this allows practitioners (for example, those working on network optimization applied in Systems Biology, where large-scale instances are usually common) without access to an ILP-solver, to have at hand an approach which delivers good quality primal solutions (along with accurate dual bounds). Note that the designed R&C framework is provided for download at [Sinnl and Álvarez-Miranda].

An important feature of the devised scheme is its flexibility to easily adapt to different variants of the MWCS; namely, different structures of the side constraints embodied by $\Lambda \mathbf{y} \leq \boldsymbol{\beta}$, e.g., degree constraints, diameter constraints, quota constraints, etc. Furthermore, the designed R&C algorithm can be generalized to other related problems such as uniform-weight Steiner tree problems or the node-weighted connected dominating set [see, e.g. Bley et al.]. This is certainly an interesting path for future work.

References

11th DIMACS Implementation Challenge. Steiner tree problems, 2014. URL `http://dimacs11.cs.princeton.edu/home.html`.

A. Aboudi, R. Hallefjord and K. Jörnsten. A facet generation and relaxation technique applied to an assignment problem with side constraints. *European Journal of Operational Research*, 50, 1991.

N. Adluru, X. Yang, and L. Latecki. Sequential monte carlo for maximum weight subgraphs with application to solving image jigsaw puzzles. *International Journal of Computer Vision*, pages 1–23, 2014.

M. Akhmedov, I. Kwee, and R. Montemanni. A divide and conquer matheuristic algorithm for the prize-collecting steiner tree problem. *Computers & Operations Research*, 70:18–25, 2016.

R. Albert. Scale-free networks in cell biology. *Journal of Cell Science*, 118(21):4947–4957, 2005.

E. Althaus and M. Blumenstock. Algorithms for the maximum weight connected subgraph and prize-collecting steiner tree problems. *Workshop of the 11th DIMACS Implementation Challenge*, 2014.

E. Álvarez-Miranda, I. Ljubić, and P. Mutzel. The maximum weight connected subgraph problem. In M. Jünger and G. Reinelt, editors, *Facets of Combinatorial Optimization*, pages 245–270. Springer, 2013a.

E. Álvarez-Miranda, I. Ljubić, and P. Mutzel. The rooted maximum node-weight connected subgraph problem. In C. Gomes and M. Sellmann, editors, *Proceedings of CPAIOR 2013*, volume 7874 of *LNCS*, pages 300–315. Springer, 2013b.

S. Andreotti. *Linear Programming and Integer Linear Programming in Bioinformatics*. PhD thesis, The Ohio State University, February 2015.

K. Anstreicher and L. Wolsey. Two "well-known" properties of subgradient optimization. *Mathematical Programming*, 120(1):213–220, 2007.

C. Backes, A. Rurainski, G. Klau, O. Müller, D. Stöckel, A. Gerasch, J. Küntzer, D. Maisel, N. Ludwig, M. Hein, A. Keller, H. Burtscher, M. Kaufmann, E. Meese, and H. Lenhof. An integer linear programming approach for finding deregulated subgraphs in regulatory networks. *Nucleic Acids Research*, 1:1–13, 2011.

J. Beasley. Lagrangian relaxation. In *Modern heuristic techniques for combinatorial problems*, pages 243–303. John Wiley & Sons, 1993.

D. Beisser, G. Klau, T. Dandekar, T. Müller, and M. Dittrich. BioNet: An R-package for the functional analysis of biological networks. *Bioinformatics*, 26(8):1129–1130, 2010.

A. Bley, I. Ljubić, and O. Maurer. A node-based ilp formulation for the node-weighted dominating steiner problem. *Technical Report ISOR, University of Vienna*. URL `http://homepage.univie.ac.at/ivana.ljubic/research/publications/nwst.pdf`.

R. Carvajal, M. Constantino, M. Goycoolea, J.P. Vielma, and A. Weintraub. Imposing connectivity constraints in forest planning models. *Operations Research*, 61(4):824–836, 2013.

C. Chen and K. Grauman. Efficient activity detection with max-subgraph search. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition 2012*, pages 1274–1281. IEEE, 2012.

G. Csardi and T. Nepusz. The igraph software package for complex network research. *InterJournal, Complex Systems*, 1695(5):1–9, 2006.

A. da Cunha, A. Lucena, N. Maculan, and M. Resende. A relax-and-cut algorithm for the prize-collecting steiner problem in graphs. *Discrete Applied Mathematics*, 157(6):1198–1217, 2009.

M. de Aragão and R. Werneck. On the implementation of mst-based heuristics for the steiner problem in graphs. In *Algorithm engineering and experiments*, pages 1–15. Springer, 2002.

B. Dilkina and C. Gomes. Solving connected subgraph problems in wildlife conservation. In A. Lodi, M. Milano, and P. Toth, editors, *Proceedings of CPAIOR 2010*, volume 6140 of *LNCS*, pages 102–116. Springer, 2010.

M. Dittrich, G. Klau, A. Rosenwald, T. Dandekar, and T. Muller. Identifying functional modules in protein-protein interaction networks: an integrated exact approach. *Bioinformatics*, 24(13):i223–i231, 2008.

M. El-Kebir. *Networks, modules and breeding schedules: Applications of Combinatorial Optimization to Computational Biology*. PhD thesis, Vrije Universiteit, October 2015.

M. El-Kebir and G. Klau. Solving the maximum-weight connected subgraph problem to optimality. *Workshop of the 11th DIMACS Implementation Challenge*, 2014.

S. Engevall, M. Göthe-Lundgren, and P. Värbrand. A strong lower bound for the node weighted steiner tree problem. *Networks*, 31(1):11–17, 1998.

L. Escudero, M. Guignard, and K. Malik. A lagrangian relax-and-cut approach for the sequential ordering problem with precedence relationships. *Annals of Operations Research*, 50(1):219–237, 1994.

M. Fischetti, M. Leitner, I. Ljubić, M. Luipersbeck, M. Monaci, M. Resch, D. Salvagnin, and M. Sinnl. Thinning out steiner trees a node-based model for uniform edge costs. *Workshop of the 11th DIMACS Implementation Challenge*, 2014.

C. Friedel and R. Zimmer. Toward the complete interactome. *Nature Biotechnology*, 24(6):614–615, 2006.

M. Haouari, S. Layeb, and H. Sherali. The prize collecting steiner tree problem: models and lagrangian dual optimization approaches. *Computational Optimization and Applications*, 40(1):13–39, 2008.

M. Haouari, S. Layeb, and H. Sherali. Algorithmic expedients for the prize collecting Steiner tree problem. *Discrete Optimization*, 7(1-2):32–47, 2010.

A. Hatem. *Active Module Discovery: Integrated Approaches of Gene Co-Expression and PPI Networks and MicroRNA Data*. PhD thesis, The Ohio State University, 2014.

C. Hegde, P. Indyk, and L. Schmidt. A fast, adaptive variant of the goemans-williamson scheme for the prize-collecting steiner tree problem. *Workshop of the 11th DIMACS Implementation Challenge*. URL `https://people.csail.mit.edu/ludwigs/papers/dimacs14_fastpcst.pdf`.

S. Huang. *A constraint optimization framework for discovery of cellular signaling and regulatory networks*. PhD thesis, Massachusetts Institute of Technology, June 2011.

T. Ideker, O. Ozier, B. Schwikowski, and A. Siegel. Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics*, 18(Supplement 1):S233–S240, 2002.

V. Janjić, R. Sharan, and N. Przulj. Modelling the yeast interactome. *Scientific Reports*, 2008.

D. Johnson, M. Minkoff, and S. Phillips. The prize collecting steiner tree problem: theory and practice. In D. Shmoys, editor, *Proceedings of the 11th Symposium on Discrete Algorithms*, pages 760–769. ACM/SIAM, 2000.

H. Kellerer, U. Pferschy, and D. Pisinger, editors. *Knapsack Problems*. Springer, 1st edition, 2004.

K. Kiwiel, T. Larsson, and P. Lindberg. Lagrangian relaxation via ballstep subgradient methods. *Mathematics of Operations Research*, 32(3):669–686, 2007.

A. Klose. A lagrangean relax-and-cut approach for the two-stage capacitated facility location problem. *European Journal of Operational Research*, 126(2):408–421, 2000.

T. Kuo, K. Lin, and M. Tsai. Maximizing submodular set function with connectivity constraint: Theory and application to networks. *IEEE/ACM Transactions on Networking*, 23(2):533–546, 2015.

I. Ljubić, R. Weiskircher, U. Pferschy, G. Klau, P. Mutzel, and M. Fischetti. An algorithmic framework for the exact solution of the prize-collecting Steiner tree problem. *Mathematical Progamming*, Series B(105): 427–449, 2006.

A. Lucena. Non delayed relax-and-cut algorithms. *Annals of Operations Research*, 140(1):375–410, 2005.

A. Lucena. Lagrangian relax-and-cut algorithms. In M. Resende and P. Pardalos, editors, *Handbook of Optimization in Telecommunications*. Springer, 2006.

Guignard. M. Efficient cuts in lagrangean relax-and-cut schemes. *European Journal of Operational Research*, 105, 1998.

S. Martello and P. Toth. *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc., 1990.

S. Martello, D. Pisinger, and P. Toth. Dynamic programming and strong bounds for the 0-1 knapsack problem. *Management Science*, 45(3):414–424, 1999.

Y. Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120(1): 221–259, 2007.

N. Przulj, D. Corneil, and I. Jurisica. Modeling interactome: scale-free or geometric? *Bioinformatics*, 20 (18):3508–3515, 2004.

D. Rajagopalan and P. Agarwal. Inferring pathways from gene lists using a literature-derived network of biological relationships. *Bioinformatics*, 21(6):788–793, 2005.

H. Sherali and G. Choi. Recovery of primal solutions when using subgradient optimization methods to solve lagrangian duals of linear programs. *Operations Research Letters*, 19(3):105–113, 1996.

H. Sherali and O. Ulular. Conjugate gradient methods using quasi-newton updates with inexact line searches. *Journal of Mathematical Analysis and Applications*, 150(2):359–377, 1990.

M. Sinnl and E. Álvarez-Miranda, E. RCMWCS - a relax-and-cut based MWCS solver. `http://homepage.univie.ac.at/markus.sinnl/program-codes/LMWCS`.

S. Vijayanarasimhan and K. Grauman. Efficient region search for object detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition 2011*, pages 1401–1408. IEEE, 2011.

Y. Wang, A. Buchanan, and S. Butenko. On imposing connectivity constraints in integer programs. *Optimization Online*, 2014.

L. Wolsey. *Integer Programming*. Wiley, 1998.

T. Yamamoto, H. Bannai, M. Nagasaki, and S. Miyano. Better decomposition heuristics for the maximum-weight connected graph problem using betweenness centrality. In J. Gama, V. Costa, A. Jorge, and P. Brazdil, editors, *Discovery Science*, volume 5808 of *LNCS*, pages 465–472. Springer, 2009.

N. Yosef, E. Zalckvar, A. Rubinstein, M. Homilius, N. Atias, L. Vardi, I. Berman, H. Zur, A. Kimchi, E. Ruppin, and R. Sharan. ANAT: A tool for constructing and analyzing functional protein networks. *Science Signaling*, 4(196):pl1–pl1, 2011.