Pontifícia Universidade Católica
do Rio de Janeiro

Larissa Figueiredo Terra de Faria

The Multi-Period Prize-Collecting Steiner Tree Problem with Budget Constraints

Tese de Doutorado

Thesis presented to the Programa de Pós-graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Ciências - Informática.

Advisor : Prof. Hélio Côrtes Vieira Lopes
Co-advisor: Dr. David Sotelo Pinheiro da Silva

Rio de Janeiro
April 2019

Pontifícia Universidade Católica
do Rio de Janeiro

Larissa Figueiredo Terra de Faria

The Multi-Period Prize-Collecting Steiner Tree Problem with Budget Constraints

Thesis presented to the Programa de Pós-graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Ciências - Informática. Approved by the undersigned Examination Committee.

Prof. Hélio Côrtes Vieira Lopes
Advisor
Departamento de Informática – PUC-Rio

Dr. David Sotelo Pinheiro da Silva
Co-advisor
Pesquisa Operacional e Ciência de Dados – Petrobras

Prof. Marcus Vinicius Soledade Poggi de Aragao
Departamento de Informática – PUC-Rio

Dr. Luiz Carlos Ferreira de Sousa
Pesquisa Operacional e Ciência de Dados – Petrobras

Prof. Eduardo Uchôa Barboza
Departamento de Engenharia de Produção – UFF

Prof. Rafael Martinelli Pinto
Departamento de Engenharia Industrial – PUC-Rio

Rio de Janeiro, April 5th, 2019

Larissa Figueiredo Terra de Faria

Graduated in Industrial and Electrical Engineering from PUC-Rio in 2008, specializing in Decision Support Systems. During that time, completed an academic exchange program at INSA-Lyon. Holds a M.Sc. degree in Electrical Engineering, also from PUC-Rio. Currently works as an Operations Research Analyst at Petrobras, developing and implementing decision support models, that commonly use simulation, optimization and/or prediction methods. Completed a research internship at CIRRELT in Montréal, participating in a doctorate academic exchange program.

Bibliographic data

Faria, Larissa Figueiredo Terra de

The Multi-Period Prize-Collecting Steiner Tree Problem with Budget Constraints / Larissa Figueiredo Terra de Faria; advisor: Hélio Côrtes Vieira Lopes; co-advisor: David Sotelo Pinheiro da Silva. – Rio de Janeiro: PUC-Rio, Departamento de Informática, 2019.

119 f: il. color. ; 29,7 cm

Tese (doutorado) – Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2019.

Inclui bibliografia.

1. Informática – Teses. 2. Árvore de Steiner com coleta de prêmios. 3. Multi-período. 4. Branch-and-Cut. 5. Desenho de Rede. I. Lopes, Hélio Côrtes Vieira. II. Silva, David Sotelo Pinheiro da. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. IV. Título.

CDD: 004

To my parents, Cynthia Faria and Luiz Paulo Faria.
To my brother, Lucas Faria and to my sister-in-law, Carol Rodriguez.

You are everything to me.

Acknowledgments

To David Sotelo, whom I greatly admire as a researcher, teacher, consultant, operations research expert, systems analyst developer, data science enthusiast, and finally for the empathetic and sensible human being that he is, for helping me endlessly along this journey, from the moment he first encouraged me to pursue this, to the very last day of work, giving me numerous advice, not only about algorithms, but also about emotional balance, never allowing me to doubt myself and always pushing me forward;

To Hélio Lopes, for accepting me in the Computer Sciences program, giving me the opportunity to learn remarkably so about algorithms, object-oriented programming, data science and so much more;

To Luiz Carlos Costa, for valuing the importance of research in our every day work, for supporting the opportunity I was given to study abroad, allowing my professional and personal growth, and finally for helping me in every way upon my return, guaranteeing the success of this endeavor;

To Sanjay Jena and Jean François Cordeau, for welcoming me into the CIRRELT lab, for making my research theirs, contributing greatly along the year I was in Canada and for maintaining that partnership moving forward;

To the members of the Examination Committee, for their extremely valuable contributions;

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) Finance Code 001

To the friends Daniel Fuller, Gabriel Lisbôa and Jacques Brawerman, not only for helping me learn Java programming, keep the server stable and automatize the runs, but also for creating such a wonderfully harmonious environment at work;

To the friends Aldair Álvarez, Amira Dems, Bruno Bruck, Luciano Costa, Narges Sereshti, Rennan Chagas and Renan Butkeraites, for helping me with the mathematical programming model, for teaching me all there is to know about CPLEX and for sharing side by side the amazing experience to live and learn abroad;

To the friends Arthur Santos, Clarissa Gandour, Jan Siqueira, Lívia Gama, Lize Dias, Paula Nunes and Vinícius Costa, for giving me certainty in a world of uncertainty, being there no matter what, when or where;

To my family, for loving me unconditionally and always telling me I have the capacity to accomplish whatever I set my mind to, as I ended up believing it;

To Christophe Muzeau, for setting up a server from scratch just for me, but mostly for saving me from myself, for giving me all the emotional support, kindness, patience and love in this world.

I am forever grateful.

Abstract

Faria, Larissa Figueiredo Terra de; Lopes, Hélio Côrtes Vieira (Advisor); Silva, David Sotelo Pinheiro da (Co-Advisor). **The Multi-Period Prize-Collecting Steiner Tree Problem with Budget Constraints**. Rio de Janeiro, 2019. 119p. Tese de doutorado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

This thesis generalizes the multi-period variant of the classical Prize-collecting Steiner Tree Problem, which aims at finding a connected subgraph that maximizes the revenues collected from connected nodes minus the costs to utilize the connecting edges. This work additionally: (a) allows vertices to be added to the tree at different time periods; (b) imposes a predefined budget on edges selected over a specific horizon of time periods; and (c) limits the total length of edges that can be added over a time period. A *branch-and-cut* algorithm is provided for this problem, satisfactorily evaluating benchmark instances from the literature, adapted to a multi-period setting, up to approximately 2000 vertices and 200 terminals in reasonable time.

Keywords

Prize-Collecting Steiner Tree; Multi-period; Branch-and-Cut; Network Design.

Resumo

Faria, Larissa Figueiredo Terra de; Lopes, Hélio Côrtes Vieira; Silva, David Sotelo Pinheiro da. **O Problema Multi-Período da Árvore de Steiner com coletas de prêmios e restrições de orçamento**. Rio de Janeiro, 2019. 119p. Tese de Doutorado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Esta tese generaliza a variante multi-período do clássico problema da Árvore de Steiner com coleta de prêmios (PCST), que visa encontrar um subgrafo conexo que maximize os prêmios recuperados de nós conectados menos o custo de utilização das arestas conectadas. Este trabalho adicionalmente: (a) permite que vértices sejam conectados à árvore em diferentes períodos de tempo; (b) impõe um orçamento pré-definido em arestas selecionadas em um horizonte específico de períodos de tempo; e (c) limita o comprimento total de arestas que podem ser adicionadas em um período de tempo. Um algoritmo *branch-and-cut* é fornecido para este problema, avaliando satisfatoriamente instâncias *benchmark* da literatura, adaptadas para uma configuração multi-período, de até aproximadamente 2000 vértices e 200 terminais em tempo razoável.

Palavras-chave

Árvore de Steiner com coleta de prêmios; Multi-período; Branch-and-Cut; Desenho de Rede.

Table of Contents

List of Figures

List of Tables

"For me, I am driven by two main philosophies, know more today about the world than I knew yesterday. And along the way, lessen the suffering of others. You'd be surprised how far that gets you."

Neil deGrasse Tyson, *I am Neil deGrasse Tyson - AMA.*

Chapter 1
Introduction

1.1
Motivation: Natural Gas Network Expansion Problem

The natural gas industry, as well as other infrastructure sectors, is an example of networking industry. This type of industry is characterized by the presence of distinct activities constituted in the form of a physical network, in which the interconnection is essential to its operation and provision of the service. The gas network is composed of pipelines, that connect one city to the next, provisioning natural gas. Given a distribution center, represented as the root of the pipeline network, a city is said connected to the pipeline network if there is a pipeline path from the distribution center to this city. Also, to ensure a good prediction of future demand for natural gas, one must study the distribution network of the gas pipeline, which is dynamic, and learn to estimate its future expansion.

Many countries do not possess a complete gas pipeline network. That is, one that extends to all the cities in their territories. This situation is specially worse for third-world countries, where most cities lack the distribution pipeline infrastructure and have to resort to compressed natural gas (CNG), delivered by trucks, to supply their demand for the resource. Developing countries naturally aim for greater rates of economic growth and such rates are closely linked to building energy distribution infrastructure. Therefore the natural gas network expansion problem can be considered a world-wide problem.

A typical planning scenario has as its input a set of potential customers, together with the discounted future profits they would generate, and also a potential network. In the case specified by this thesis, the potential costumers are represented by the non-connected cities and the potential network is defined by the possibilities of connections between these cities and the existing network. The customers generate profit by consuming the natural gas provided by the network. Costs of the network are defined as the costs of building the stretches of pipeline necessary to connect a city to the network. The situation is similar for other utilities like fiber optic connections or district heating (Ljubić et al., 2005).

Essentially, we are confronted with a network design problem. Network Design Problems are a subclass of Combinatorial Optimization problems dealing with the selection of subgraphs of a given graph that preserve some predefined structural requirements. Regarding to these requirements, some possible options are preserving the entire vertex set of the original graph into the same connected component (Spanning Tree Problem) or to consider a partition of the vertex set into required and optional ones (named Steiner vertices) while asserting that all required vertices belong to the same connected component in the resulting subgraph (Steiner Tree Problem), among other possible requirement definitions (Magnanti and Raghavan, 2005).

In other words, assuming we aim to maximize overall profit, the decision process consists, on one hand, in selecting a subset of particular profitable customers; and on the other hand, in designing a network that connects all selected customers in a cost-efficient way to the existing network. Natural gas flows in both directions of a pipeline stretch, leading to the pipeline network representation as an undirected graph. The natural trade-off between maximizing the sum of profits over all selected customers and minimizing the cost of the network leads to a prize-collecting objective function (Bienstock et al., 1993). This problem is defined as the Prize-Collecting Steiner Tree problem (PCST). We can formulate it mathematically as shown in Section 1.3.

Moreover, the problem at hand involves the planning of the gas network expansion throughout a multiple number of periods in the near future. That is, the problem is studied over a specified horizon. Therefore, the decisions made in one period affect the decisions made in the next. The period in which a city is incorporated to the network results in different profits, usually related to the sum of the demands of that city, from the corresponding period to the end of the time horizon. These profits are represented as prizes, which are associated to the vertices of the undirected graph. In the same way, building a new pipeline stretch brings a new cost into play, which is associated to a corresponding edge of the undirected graph. Furthermore, it is certain that the distribution company in charge of such a network expansion has budget constraints to deal with. Budget constraints restrict, for a subset of time periods, the maximum total cost that can be spent on building pipeline stretches. There may be not only a financial budget available per subset of time periods, but also a distance limit per time period to comply with, due to physical and logistical construction restrictions.

These extra elements extend the original PCST problem. Therefore, the problem at hand is named the Multi-period Prize-Collecting Steiner

Tree problem with Budget constraints (MPCSTB). The MPCSTB is defined thoroughly in Section 1.4. The practical purpose, then, of this thesis is to plan the expansion of a gas network throughout a multiple number of periods in the near future by developing an optimization model that objectively identifies the expansion trends of the distribution network by maximizing the potential profit increment for the distributor, identifying the most attractive cities in its area.

This work is organized in the following way: Section 1.2 presents the academic contributions of this thesis, Section 1.3 presents the mathematical formulation of the PCST problem and Section 1.4 describes the MPCSTB in detail. Chapter 2 provides a review on the literature around the MPCSTB problem. Chapter 3 introduces a mixed integer linear programming formulation for it. Chapter 4 provides a detailed explanation about a *branch-and-cut* algorithm for the problem. Computational results are shown in Chapter 5. Finally, Chapter 6 draws conclusions and possible future work directions.

1.2 Contributions

We are interested in providing a *branch-and-cut* algorithm to an extension of the PCST, which is named the MPCSTB. This problem is not new, as can be seen in Suhl and Hilbert (1998). Therefore, our academic contributions are mainly related to the algorithm, not to the problem itself. We aim to obtain an exact method model that is able to solve reasonably large instances for a quite natural extension of a classical problem (previously studied in Lucena and Resende (2004); Ljubić et al. (2005, 2006); Gollowitzer and Ljubić (2011); Arulselvan et al. (2011); Fischetti et al. (2017), among others), finding solutions of guaranteed quality for realistic problem sizes in a reasonable amount of computing time. The computational experiments focus on evaluating the performance of our model and on exploring the impact of different budget limitations.

1.3 Definition: The Prize-Collecting Steiner Tree Problem

It was established in Section 1.1 that the problem of finding the optimal expansion of a country's gas network can be generalized to a network design problem. Among all network design problems, to the best of our knowledge, the one that is closest to the problem at hand is the Prize-Collecting Steiner Tree problem (PCST). The PCST defines prizes for each vertex and costs for each edge of the original graph, asking for a connected subgraph that maximizes the sum of the prizes of the selected vertices decreased by the sum of the costs of the selected edges (Ljubić et al., 2005).

Definition 1 (Prize-Collecting Steiner Tree Problem, PCST)**.** Let $G = (V, E, c, r)$ be an undirected graph, with a *revenue function* $r : V \mapsto \mathbb{Q}^+$ over its vertices and a *cost function* $c : E \mapsto \mathbb{Q}^+$ over its edges. The Prize-Collecting Steiner Tree problem (PCST) consists of finding a connected subgraph $Z = (V_Z, E_Z)$ of G, $V_Z \subseteq V$, $E_Z \subseteq E$ that maximizes

$$profit(Z) = \sum_{v \in V_Z} r(v) - \sum_{e \in E_Z} c(e). \tag{1.1}$$

It is easy to see that every optimal solution Z will be a tree. More precisely, a Steiner tree, which is defined by selecting the most profitable vertices and connecting them by a least-cost network (Ljubić et al., 2006).

PCST is NP-Hard (Karp, 1972) and the search for exact methods solving families of large instances to optimality has received a considerable amount of attention in recent years (Johnson et al., 2000; Canuto et al., 2001; Klau et al., 2004; Lucena and Resende, 2004; Ljubić et al., 2005, 2006; Feofiloff et al., 2007; da Cunha et al., 2009; Fischetti et al., 2017; Gamrath et al., 2017). In addition, the fact that PCST can be used to model a large number of real-world problems related to network expansions has motivated the study of variations of this classical problem (Suhl and Hilbert, 1998; Costa et al., 2006, 2009; Gollowitzer and Ljubić, 2011; Arulselvan et al., 2011).

In this thesis, we propose an exact method for a variation of PCST, denoted by the Multi-period Prize-Collecting Steiner Tree problem with Budget constraints (MPCSTB).

1.4 Definition: The Multi-period Prize-Collecting Steiner Tree problem with Budget constraints

This problem takes into account three additional elements when compared to the classical PCST:

1. The fact that every vertex or edge must be added to the solution at a specific time period, chosen from a discrete set called time horizon. Hence, the prize associated to the insertion of a vertex will depend on the time period that it was added to the solution.

2. The existence of budgets per set of time periods, which limit the sum of the costs of the edges that can be added at a specific horizon of time periods.

3. The total length of edges added is also limited over a time period.

In mathematical terms, the problem instance considered in this work is defined by an undirected graph $G = (V, E)$ that represents a distribution network. In this graph, the set of vertices V symbolizes the nodes to be considered, while the set of edges E corresponds to pairs of nodes which can be directly connected. There is a specially identified rooted vertex $v_0 \in V$ that represents all nodes that are already connected to the network. A function $c : (E \times \{1, \ldots, T\}) \to \mathbb{Q}^+$, denotes the cost for connecting pairs of nodes represented in E for each time period $t \in \{1, \ldots, T\}$. The total budget limit for construction of the network expansion, over a set of periods of the study horizon, is denoted by *budgetLimit*. A function $d : E \to \mathbb{Q}^+$, denotes the distance for connecting pairs of nodes represented in E. The distance limit, which is the maximum distance per time period that can be built, is denoted by *distanceLimit*. There is a number of periods nT comprising the study horizon to be considered. We also denote $\hat{T}_B = \{T_B\}$, where $T_B \subseteq T$. $\hat{T}_B$ is a set of consecutive periods that belong in T for which there is a budget limit to be respected. Finally, a function *profit* $: (V \times \{1, \ldots, T\}) \to \mathbb{Q}^+$ to represent the profit margin to be obtained if a vertex $v \in V$ is added to the network during period $t \in \{1, \ldots, T\}$.

The set of all vertices V can be divided into terminal nodes, representing vertices that have profit greater than zero, and Steiner nodes, that represent vertices that have profit equal to zero. The expected result of the problem is the estimated time period for the network to reach each node, within the horizon of the proposed study. Hence the problem output is a connected subgraph $Z_t = (V_{Z_t}, E_{Z_t})$ of G, $V_Z \subseteq V$, $E_Z \subseteq E$ for each period of the study horizon. More precisely, the functions $\alpha : V_Z \to T$ and $\beta : E_Z \to T$ map the vertices and edges of Z to the time horizon, giving the output subgraph $Z_t = (V_{Z_t}, E_{Z_t})$ of Z where $V_{Z_t} = \{v \in V_Z \mid \alpha(v) \leq t\}$ and $E_{Z_t} = \{e \in E_Z \mid \beta(e) \leq t\}$. It is important to note that this result is also relevant to the development of effective studies that seek to estimate the demand for natural gas.

Definition 2 (Multi-period Prize-Collecting Steiner Tree Problem with Budget Constraints, MPCSTB)**.** Let $T = \{1, \ldots, |T|\}$ be a *time horizon* over which is defined a function *distanceLimit* $: T \to \mathbb{Q}^+$. Let $\hat{T}_B = \{T_B\}$, where $T_B \subseteq T$ be a subset of the time horizon over which is defined a function *budgetLimit* $: \hat{T}_B \to \mathbb{Q}^+$. Let $G = (V, E)$ be an undirected graph with a *revenue function* over its vertices defined as $r : (V, T) \to \mathbb{Q}^+$, a *cost function* over its edges defined as $c : (E, T) \to \mathbb{Q}^+$ and a *distance function* over its edges defined as $d : E \to \mathbb{Q}^+$. Furthermore, let $Z = (V_Z, E_Z)$ be a subgraph of G with functions $\alpha : V_Z \to T$ and $\beta : E_Z \to T$ mapping its vertices and edges to the time horizon. Finally, let $Z_t = (V_{Z_t}, E_{Z_t})$ be the subgraph of Z where $V_{Z_t} = \{v \in V_Z \mid \alpha(v) \leq t\}$ and $E_{Z_t} = \{e \in E_Z \mid \beta(e) \leq t\}$. The `Multi-period Prize-Collecting Steiner Tree problem with Budget constraints` (MPCSTB) consists of finding a subgraph Z and the corresponding functions α and β, which maximizes:

$$profit(Z) = \sum_{v \in V_Z} r(v, \alpha(v)) - \sum_{e \in E_Z} c(e, \beta(e)) \tag{1.2}$$

subject to

$$\sum_{e \in E_{Z_t}} c(e, \beta(e)) \leq budgetLimit_{T_B},\ \forall T_B \in \hat{T}_B \tag{1.3}$$

$$\sum_{e \in E_{Z_t} \setminus E_{Z_(t-1)}} d(e) \leq distanceLimit_t,\ \forall t \in T \tag{1.4}$$

and Z_t is connected.

Figure 1.1 illustrates an example of a MPCSTB instance (based on a PCST instance given in Ljubić et al. (2006)) with three time periods. Each edge has fixed costs and a length marked in kilometers, hollow circles represent terminal nodes and filled circles represent Steiner nodes. Each time period has a distance limit of 11 kilometers. The three-period time horizon has a budget limit of 100 cost units. Figure 1.2 shows a feasible solution for the first period of the study horizon, Figure 1.3 shows a feasible solution for the second period and Figure 1.4 shows the final feasible, but not optimal, solution (all three periods).

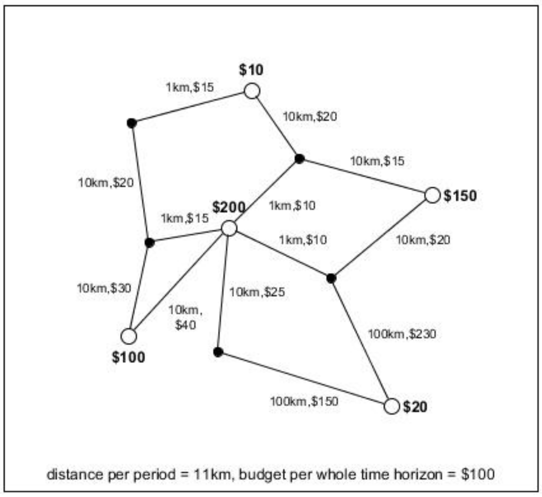


Figure 1.1: Example of a MPCSTB instance.

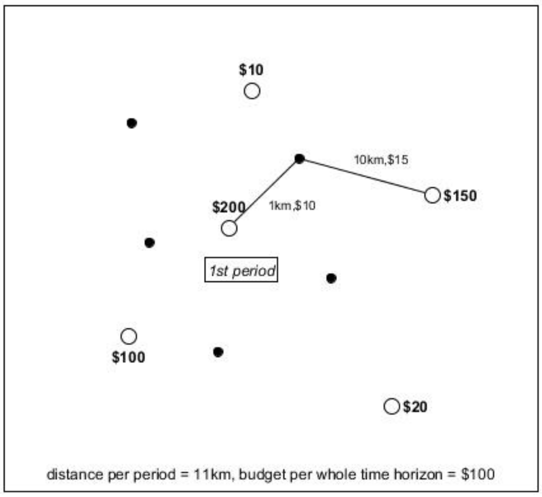


Figure 1.2: First period feasible solution.

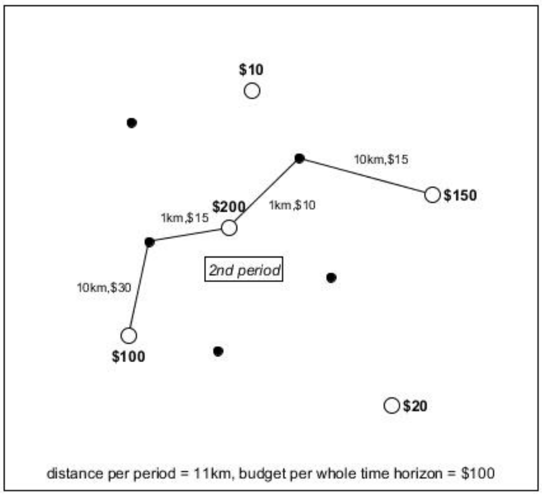


Figure 1.3: Second period feasible solution.

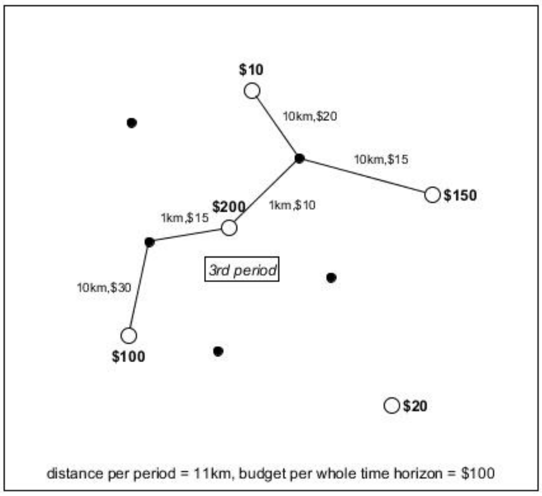


Figure 1.4: Feasible, but not optimal solution of MPCSTB.

Chapter 2
Literature Review

2.1
Network design problems

Natural gas planning problems have been of interest for the last few decades. Among them, lies the problem at hand, which hopes to find the optimal expansion of gas pipeline systems. As we have seen in Chapter 1, the MPCSTB is closely related to network design problems. Literature defines a network design problem as a problem that involves identifying a subset of edges in a graph satisfying a set of constraints with minimum total weights (or costs) (Pahl et al., 2011). Algorithms for these problems can be usually classified into two major fronts of solution methods: exact approaches and heuristics. As such problems are combinatorial and NP-hard in nature, typically a combination of both fronts are used to solve them. Exact techniques include cutting planes and branch-and-bound and they are often used combined to a variety of commercial and open source solvers (Borraz-Sánchez et al., 2016). The modeling of a network design problem may involve its operation, its expansion or both. Also, flow variables and flow-related constraints may be added to the problem, depending on its purpose (Kabirian and Hemmati, 2007). There is a considerable number of works in the literature that are interested in finding the optimal solution that would capture physical, operational or even contractual constraints (Borraz-Sánchez et al., 2016; Hansen et al., 1991; Soliman and Murtagh, 1982; De Wolf and Smeers, 1996; De Wolf and Bakhouya, 2012; Babonneau et al., 2012; Elshiekh et al., 2013; Üster and Dilaveroğlu, 2014; Humpola and Fügenschuh, 2015; Humpola et al., 2016; Atamtürk, 2002; Poss, 2012). Some of those works focus in adjusting the network's parameters, including models of regular pipelines, valves, short pipes, control valves, compressor stations, and regulators, instead of focusing solely on the network's expansion, as we intend to do.

2.2 Prize-Collecting Steiner Tree problems

As we have established in Chapter 1, the closest network design problem to our own we could find in the literature, to the best of our knowledge, is the MPCSTB, which is an extension of the PCST. Bienstock et al. (1993) introduced the PCST and developed a factor 3 approximation algorithm for it. Other approximation algorithms have been presented along the literature (Ljubić et al., 2005). Goemans and Williamson (1997) followed, proposing an approximation algorithm that yields solutions within a factor of $2 - (1/(n-1))$ of optimality and that runs in $O(n^3 \log n)$ time ($n := |V|$). Their result has been improved in Johnson et al. (2000), with the proposition of a $2 - (1/(n-1))$–approximation algorithm with $O(n^2 \log n)$ running time. Afterwards, Feofiloff et al. (2007) developed an algorithm that achieves a ratio of $(2 - 2/n)$ within the same time.

Lucena and Resende (2004) focus in presenting an integer programming formulation of the PCST and the authors are able to describe an algorithm based on polyhedral cutting planes to obtain lower bounds for the problem. The study of algorithms to solve the PCST continues throughout the literature and Ljubić et al. (2005) construct a *branch-and-cut* algorithm based on a directed graph model where they manage to efficiently separate sets of violated inequalities using a maximum flow algorithm. Moreover, Ljubić et al. (2006) have aimed to solve large and difficult instances of PCST to optimality within reasonable running time. They build a *branch-and-cut* algorithm that relies on connectivity inequalities inserted on the fly as cuts between an artificial root and every selected customer vertex. Costa et al. (2006) developed a survey which presents an overview of the methods developed to solve the PCST along the literature to that point.

The authors in da Cunha et al. (2009) generate primal and dual bounds to the PCST problem, by means of a Lagrangian Non Delayed Relax and Cut (NDRC) algorithm, which is capable of adequately dealing with the exponentially many candidate inequalities to dualize. Furthermore, metaheuristic approaches were also developed to attempt to solve the PCST. Canuto et al. (2001) developed a multi-start local-search-based algorithm with perturbations, while a memetic algorithm with incorporated local improvement has been presented by Klau et al. (2004). On the other hand, Uchoa (2006)'s approach to the PCST is to apply redefined reduction tests, proven to be effective on Steiner Problem in Graphs (SPG). The concept of bottleneck Steiner distance is properly redefined for the PCST.

It can be seen that the PCST is a challenging NP-hard problem. Even so, Fischetti et al. (2017) present a simple solution method and succeed to obtain very good (sometimes proven optimal) solutions for hard instances from the literature. They achieve this feat by avoiding over-modeling and focusing on a model that only has node variables, which proves to be successful for instances where all edges have the same cost.

2.3
Prize-Collecting Steiner Tree problems with budget constraints

Variations of the PCST, such as quota and budget versions of the problem, were studied in the literature. Johnson et al. (2000) define the quota version of the PCST by the search for the tree with minimum edge cost that contains vertices whose total prize is at least a given quota. Additionally, Johnson et al. (2000) consider the problem of looking for the tree with maximum prize, given that the total edge cost is within a given budget, hence, the PCST with budget constraints. Johnson et al. (2000) define the quota problem as a generalization of the k-MST problem and propose to extend constant-factor approximation algorithms to attempt to solve it. For the (unrooted) budget problem, the authors show how a $(5 + \epsilon)$-approximation algorithm can be derived from Garg's 3-approximation algorithm for the k-MST. Furthermore, Johnson et al. (2000) generalize the approach on their budget problem and propose to incorporate it into a practical heuristic, involving the performance of multiple runs of the Goemans-Williamson algorithm (Goemans and Williamson, 1997) and the use of an increasing sequence of prize multipliers.

Costa et al. (2009) define the Steiner tree problem with revenues, budget and hop constraints. This problem is a variant of the PCST problem with additional budget and hop constraints. Budget constraints impose limits on the total cost of the network, whereas hop constraints impose limits on the number of edges between a vertex and the root. The authors solve to optimality instances with up to 500 vertices and 625 edges making use of two *branch-and-cut* algorithms. When problems of that format (PCST with budget constraints) are considered, Costa et al. (2009) show that *branch-and-cut* algorithms that make use of cut constraints (instead of GSECs - generalized subtour elimination constraints) obtain the best results to date. Also, for several variants of the Steiner Tree problem (STP), directed models are proven better than their undirected counterpart, as a number of studies have attested through the literature (see, for example, Chopra and Rao (1994a), Chopra and Rao (1994b), Feremans et al. (2002), Ljubić et al. (2005) and Magnanti and Raghavan (2005)).

2.4 Other multi-period network design problems

It is important to notice that the problem at hand is a multi-period problem, so it is relevant to search for multi-period works that resembles the one we are trying to solve.

The authors in Kawatra and Bricker (2000) attempted to solve the Multi-period Capacitated Minimal Spanning Tree (MCMST) problem, that consists of minimizing the present value of expenditures with the scheduling of installation of network links so as to connect a set of terminal nodes to a central node. There is a limit of link capacities to the number of terminal nodes sharing a link. Some terminal nodes may be activated over time. This problem is formulated as an integer programming problem. Not only a branch exchange heuristic procedure is proposed to solve the problem, but also, a Lagrangian relaxation method is presented to find a lower bound for the optimal objective function value.

Chagas and Cunha (2016) consider the Multi-period Minimum Spanning Tree Problem (MMST). This problem consists in scheduling when edges are added to a connected and undirected graph, considering a finite discrete time horizon. For each time period, the partial solution must be a tree. No edge already added to the partial solution may be removed at a following time period. The final and complete solution must be a spanning tree of the original undirected graph. There are pre-defined dates for each vertex and so, the vertices spanned by these trees cannot exceed such dates. Edges' costs are obtained by the sum of the installation costs at the time period of installation plus maintenance costs, from that time period until the end of the study horizon. The purpose is to minimize the cost of the final spanning tree. The complexity of MMST is addressed in Chagas and Cunha (2016) for the first time and the authors show that, unlike the Minimum Spanning Tree (MST) case, MMST is NP-Complete.

In their following work (Chagas et al., 2018), the Multi-period Degree Constrained Minimum Spanning Tree Problem (MP-DCMSTP) is defined. One must optimize the scheduling of edges' installation over the study horizon, guaranteeing the partial solution is a degree constrained spanning tree at all times. It is important to assure that vertices are connected to a root node no later than their latest installation dates. Chagas et al. (2018) present a new integer programming formulation for the MP-DCMSTP that is at least as good as the most successful multi-commodity flow formulation in the literature to date. New valid inequalities introduced guarantee the authors' strengthened formulation to produce strong known bounds. Two

MP-DCMSTP exact algorithms exploring the strengthened formulation were proposed.

Arulselvan et al. (2011) consider the incremental connected facility location problem. This problem is defined as the optimal selection per time period of a set of facilities to open, a set of costumers to be served, and the corresponding assignment of said customers to said facilities. Also, a network connecting the open facilities must be established. As input data, it is given a set of potential facilities, a set of interconnection nodes, a set of customers with demands, and a study horizon. If a customer is served in a certain time period, it must also be served in subsequent periods. Additionally, there is a minimum coverage requirement that must be respected for each time period. The goal is to maximize the difference between the discounted revenues of serving the customers and the discounted investments and maintenance costs for the facilities and the network. The authors propose a *branch-and-cut* algorithm for this problem, after a study of different MIP models and a discussion of some valid inequalities to strengthen their formulations.

2.5 Multi-period Prize-Collecting Steiner Tree problems with Budget constraints

Finally, Suhl and Hilbert (1998) is the only article in the literature, to the best of our knowledge, that attempts to solve the MPCSTB. A gas network is represented by an undirected graph. Vertices' profits are represented by negative edge weights. Part of the graph may already be piped in previous periods and on every subsequent period, as the piped subgraph is extended, the solution must be a tree. The task is to maximize the profit obtained by connecting vertices to the network over a multi-period study horizon. Furthermore, budget and distance constraints restrict the number of node connections per period. An integer programming formulation leading to a *branch-and-cut* algorithm is used, along with an optimization software system for solving the large-scale linear problem (LP).

Our idea is solving the Multi-period Prize-Collecting Steiner Tree problem with Budget constraints using *branch-and-cut* with two separation procedures. One to separate integer solutions and the other to separate fractional ones. In addition, we apply an algorithm to tighten the upper bound of the problem, helping achieve optimality. The mathematical formulation is thoroughly shown in Chapter 3 and the solving method is specified in detail in Chapter 4.

Chapter 3
Mathematical Formulation

We present an integer programming formulation for the MPCSTB. Even though the MPCSTB assumes that edges are undirected, we will provide a formulation that is based on directed arcs, given that those formulations have shown to provide stronger linear programming relaxation (LR) bounds (as seen in Fischetti (1991); Goemans and Myung (1993); Chopra and Rao (1994a,b); Magnanti and Wolsey (1995); Feremans et al. (2002); Ljubić et al. (2005); Magnanti and Raghavan (2005)). Therefore, let $G = (V, A)$ be a directed graph with vertex set $V = \{0, ..., n\}$ and arc set $A = \{a = (i, j) : i, j \in V\}$, where each arc $a \in A$ has an associated cost c_a^t, depending on the time period considered. For $W \subseteq V$, define A_W as the set of arcs with both endpoints in W.

We denote by T the planning horizon (for example, 2019 to 2024), composed of time periods $t \in T$ (which may, for example, represent one year each). We also denote $\hat{T}_B = \{T_B\}$. T_B is a subset of T, a consecutive horizon of time periods that is ultimately in T. We assume that there is a rooted vertex, denoted as $v_0 \in V$ that represents all nodes that are already connected to the network at the beginning of the planning horizon. This rooted vertex v_0 is assumed to be available throughout all time periods. In order to guarantee that the final network is connected, one needs to ensure that all selected vertices are connected to the root node v_0. To this end, we denote $\delta^-(W) := \{(i, j) \in A | i \notin W, j \in W\}, \forall W \subseteq V$. If the instance does not have an actual root node, an artificial root node will be created for it.

We introduce binary variables $y_i^t \in \{0, 1\}, \forall i \in V, \forall t \in T$, which take value 1 if vertex i is connected for the first time in time period t, and 0 otherwise. We also use binary arc variables $x_{ij}^t \in \{0, 1\}, \forall a \in A, \forall t \in T$, which take value 1 if arc a is connected for the first time in time period t, and 0 otherwise. Constants c_{ij}^t denote the costs to install arc (i, j) in the beginning of time period t. The arc connecting costs need to be payed only once: at the time period they are built. However, note that this may also include maintenance costs for the following time periods. Constants r_i^t represent the revenues collected when selecting vertex i in the period t, therefore connecting it to the existing network. Note that the revenue constant may also contain revenues from subsequent time periods. Constants d_{ij} denote the distances between vertex i and vertex j. In summary, there is a revenue function over the vertices and a cost function over the edges.

The profit function 1.2 given in Section 1.4 is known in the literature as a function describing the *Net Worth Maximization* Problem (Johnson et al., 2000). In the so-called *Goemans and Williamson Minimization* Problem (Goemans and Williamson, 1997), the goal is to find a subtree that minimizes the objective function $\sum_T (\sum_{v \notin V_Z} r_t(v) + \sum_{e \in A_Z} c_t(a))$. Those two formulations are equivalent, as far as optimization is concerned, as can be shown in Lemma 2. Before the proof is presented, it is important to remember the following LP Lemma.

Lemma 1 (Rewriting an objective function)**.** *To change a maximization problem to a minimization problem, multiply the objective function by -1:*

$$Max\ f(x) = -Min\ \Big(-f(x)\Big). \tag{3.1}$$

Lemma 2 (*Net Worth Maximization* Problem is equivalent to the *Goemans and Williamson Minimization* Problem)**.**

Proof. The profit function 1.2 is rewritten using Lemma 1:

$$Max\ \sum_T \left(\sum_{v \in V_Z} r_t(v) - \sum_{a \in A_Z} c_t(a) \right) = -Min\ \sum_T \left(-\sum_{v \in V_Z} r_t(v) + \sum_{a \in A_Z} c_t(a) \right). \tag{3.2}$$

It is known that the revenue of the vertices in V is the sum of the revenue of vertices in V_Z plus the sum of the revenue of the vertices not in V_Z:

$$\sum_{v \in V} r_t(v) = \sum_{v \in V_Z} r_t(v) + \sum_{v \notin V_Z} r_t(v), \forall t \in T. \tag{3.3}$$

Therefore:

$$-\sum_{v \in V_Z} r_t(v) = -\sum_{v \in V} r_t(v) + \sum_{v \notin V_Z} r_t(v), \forall t \in T. \tag{3.4}$$

Which leads to the following minimization objective function:

$$Min\ \sum_T \left(-\sum_{v \in V} r_t(v) + \sum_{v \notin V_Z} r_t(v) + \sum_{a \in A_Z} c_t(a) \right). \tag{3.5}$$

Notice that $-\sum_{v \in V} r_t(v)$ is constant as it simply adds up to minus the total revenue in V and can hence be excluded from the objective function, resuming the objective function to the Goemans and Williamson Minimization Problem:

$$Min \sum_{T} \left(\sum_{v \notin V_Z} r_t(v) + \sum_{a \in A_Z} c_t(a) \right). \tag{3.6}$$

□

In this thesis, we are going to concentrate on the *Goemans and Williamson Minimization* Problem formulation, as it has been considered in the literature before (see Goemans and Williamson (1997) and Canuto et al. (2001)). The MPCSTB problem can be formulated as follows:

$$Min \sum_{t \in T} \Big(\sum_{i \in V} r_i^t \cdot (1 - y_i^t) + \sum_{(i,j) \in A} c_{ij}^t \cdot x_{ij}^t \Big). \tag{3.7}$$

Subject to

Cut constraints

$$\sum_{t'=1}^{t} \sum_{(u,v) \in \delta^-(W)} x_{uv}^{t'} \geqslant \sum_{t'=1}^{t} y_i^{t'}, \ \forall W \subseteq V \setminus \{v_0\}, i \in W, t \in T. \tag{3.8}$$

Multi-period constraints

$$\sum_{t \in T} x_{ij}^t \leqslant 1, \ \forall (i,j) \in A. \tag{3.9}$$

$$\sum_{t \in T} y_i^t \leqslant 1, \ \forall i \in V. \tag{3.10}$$

Side constraints

$$\sum_{t \in T_B} \sum_{(i,j) \in A} c_{ij}^t \cdot x_{ij}^t \leqslant budgetLimit^{T_B}, \forall T_B \in \hat{T}_B \tag{3.11}$$

$$\sum_{(i,j) \in A} d_{ij} \cdot x_{ij}^t \leqslant distanceLimit^t, \ \forall t \in T \tag{3.12}$$

Connectivity constraints

$$\sum_{t'=1}^{t} \sum_{j \in V} x_{ji}^{t'} \geqslant \sum_{t'=1}^{t} y_i^{t'}, \ \forall i \in V \setminus \{v_0\}, \forall t \in T. \tag{3.13}$$

Variable domains

$$x_{ij}^t \in \{0,1\}, \forall (i,j) \in A, t \in T. \tag{3.14}$$

$$y_i^t \in \{0,1\}, \forall i \in V, t \in T. \tag{3.15}$$

The exponentially large constraint set (3.8) ensures that, in each time period, all network vertices are connected to the root node. It does so by creating a subset W of nodes i that does not contain the root node v_0 and, while connecting i to the network, forces that an arc from $\delta^-(W)$ is also connected, i.e., an arc that will connect the set where v_0 is present to the set W where v_0 is not present. Inequalities (3.9) and (3.10) ensure that each vertex and arc can be selected at most once throughout the planning horizon. Constraints (3.11) express the maximum budget allowed for specified subsets of time periods and constraints (3.12) limit the total length of edges that can be added over each time period. The constraint set (3.13) guarantee that every selected vertex has exactly one predecessor on its path from the root. This connectivity constraint set is commonly seen in PCST formulations (Ljubić et al., 2005, 2006; Costa et al., 2009).

To improve the efficiency of the model, the following valid inequalities were added to the formulation described.

Valid inequalities

$$\sum_{t'=1}^{t} y_i^{t'} \geqslant \sum_{t'=1}^{t} x_{ij}^{t'},\ \forall i \in V \setminus \{v_0\}, \forall (i,j) \in A, \forall t \in T. \tag{3.16}$$

$$x_{ij}^t + x_{ji}^t \leqslant 1,\ \forall i \in V \setminus \{v_0\}, \forall (i,j) \in A, \forall t \in T. \tag{3.17}$$

The valid inequalities (3.16) act as connectivity constraints, indicating that if there is an arc from i to j in the network at a certain time period, vertex i has to be connected at that time period or at a previous one. The constraints (3.17) show that every arc adjacent to a vertex in the solution tree can be oriented only in one way. They are also commonly included in PCST formulations in the literature (Ljubić et al., 2005, 2006). The valid inequalities (3.17) are a special case of the cut constraints (3.8) written in their equivalent GSEC form (Ljubić et al., 2006). Adding these inequalities, specially all at once, may enlarge the LP. Even so, as they do not have to be separated implicitly during the *branch-and-cut* algorithm, they present a speed-up that balances out the enlargement og the LP. Tests that prove the computational value of these valid inequalities are discussed in Chapter 5 and are shown in Appendix A.1, A.2 and A.3.

Since an artificial root node may be used to represent the connected network, some constraints related to it may be added to the above formulation as well. Among them, there are also symmetry constraints.

Artificial root node constraints

$$\sum_{j \in V} x_{v_0 j}^{t=1} = 1 \tag{3.18}$$

$$y_{v_0}^{t=1} = 1 \tag{3.19}$$

Symmetry constraints

$$x_{v_0 j}^{t=1} + y_i^{t=1} \leqslant 1,\ \forall (i,j) \in A \mid j > i. \tag{3.20}$$

Constraints (3.18) guarantee that only one arc is chosen between the artificial root and any other vertices. This artificial arc has cost zero and does not alter the objective function value. Likewise, the artificial root node has no revenue and so, no effect in the objective function value of the model. All the same, constraints (3.19) ensure that the artificial root node enters the connected network at time period 1, aka, the first time period of the study horizon, since the root vertex represents the previously connected network. Finally, constraints (3.20) secure that the vertex adjacent to the root is the one with the smallest index. These constraints (3.20) aim at excluding a plethora of symmetric solutions, therefore considerably reducing the solution time in a branch-and-bound framework (Gamrath et al., 2017).

The generation of the entire set of cut constraints (3.8) is a critical problem of this type of integer programming (IP) model. Depending on the size of the graph, the number of cut constraints that should be produced can be extremely large and, in that case, render it impossible to solve the corresponding IP model depicting them all. Nonetheless, the number of cut constraints (3.8) actually violated by an integer solution obtained from a reduced model (one which the constraints of type (3.8) are not present) will be very small in comparison to their total number. It is very common for combinatorial optimization problems to follow the procedure of excluding cut constraints (3.8) from the original model and creating them on the fly during a branch-and-bound algorithm. As a result, a *branch-and-cut* algorithm is the method of choice to solve this kind of optimization problem (Suhl and Hilbert, 1998; Ljubić et al., 2005, 2006; Costa et al., 2009; Gollowitzer and Ljubić, 2011; Arulselvan et al., 2011).

It is important to add that two separation procedures are used in this model: one to separate integer infeasible solutions and another to separate fractional ones, both of them based on the cut constraint set (3.8). The separation procedures exploit the fact that constraints (3.8) imply the connectivity of the root to all other selected vertices (Costa et al., 2009).

During the separation phase which is applied at each node of the branch-and-bound tree, we add constraints of type (3.8) that are violated by the current solution of the LP-relaxation problem (Ljubić et al., 2005). Further detailed information concerning the separation procedures are described in Section 4.3.

3.1 A note on Generalized Subtour Elimination Constraints versus Cut Constraints

The classical generalized subtour elimination constraints (GSECs) are used in the Dantzig-Fulkerson-Johnson (DFJ) formulation and were introduced by Dantzig et al. (1954) for the Travelling Salesman Problem (TSP). A cycle C is defined by a path of arcs originating and ending at the same node. Therefore, the GSECs may be used in a model formulation for the MPCSTB acting in a similar way as cut constraints (3.8), with the goal of the excluding cycles from the solution graph Z within a time period: no cycles C of arcs may be in the solution for period t.

$$\sum_{t \in T} \sum_{(i,j) \in C} x_{ij}^t \leq |C| - 1, \ , \forall C \in Z. \tag{3.21}$$

Consider a solution that is not connected to the root node and, instead, forms a subtour of $n_1 < n$ nodes. We note that the sum of the x_{ij}^t for those links (i, j) in the subtour is n_1. Hence we can eliminate this type of solution by imposing the condition that the sum of x_{ij}^t over all links (i, j) connecting nodes in the subset C of n_1 vertices ($|C| = n_1$) be less than n_1, i.e, Equation (3.21) (Dantzig et al., 1954).

In our chosen formulation, we use cut constraints (3.8) to guarantee connectivity in the network. They ensure that, for each and every $W \subseteq V$ that includes a vertex i, but not the root vertex v_0, at least one of the arcs in the set of all incoming arcs in W must be built if node i is connected. Note that disconnectivity would imply the existence of a cut separating v_0 and i which would clearly violate the corresponding cut constraint. These inequalities correspond to the directed cutset inequalities in the Steiner tree formulation (Arulselvan et al., 2011).

The set of constraints (3.21) is exponentially large and so, must have a similar treatment as the set of constraints (3.8). Only violated constraints obtained from a reduced model should be added on the fly through a *branch-and-cut* algorithm. As a result, all cycles C present in the integer solution Z will be eliminated.

Even though the lower bounding procedure presented in Lucena and Resende (2004) is based on undirected GSECs (3.21), Chopra and Rao (1994a) have shown for the Steiner tree problem (STP) that directed GSECs dominate directed counterparts of several other facet defining inequalities of the undirected (GSEC) formulation. This is also the reason why the directed GSEC formulation is preferable in practice (Ljubić et al., 2006). A number of studies have followed and shown that for several variants of the STP, directed models are better than their undirected counterpart (Chopra and Rao, 1994a,b; Feremans et al., 2002; Ljubić et al., 2005; Magnanti and Raghavan, 2005; Costa et al., 2009). Moreover, Fischetti (1991) show that the cut constraints (3.8) can be rewritten as a directed version of the generalized subtour elimination constraints (GSECs). On top of it all, the model chosen in this thesis that makes use of cut constraints (3.8) is less dense than his equivalent directed (GSEC) model, so it is usually computationally preferable within the *branch-and-cut* implementation (Ljubić et al., 2005) and that is why it was chosen. Chapter 5 will show how our computational results compare in practice to results that use the undirected GSECs formulation.

3.2 Previous Work Formulation

To the best of our knowledge, Suhl and Hilbert (1998) is the only work that attempts to solve the MPCSTB problem. The authors also use binary arc variables $x_a^t \in \{0,1\}, \forall a \in A, \forall t \in T$, which take value 1 if arc a is connected for the first time in time period t, and 0 otherwise. However, they do not use node variables y_i^t. They, instead, transform nodes to arcs, by creating Steiner nodes, connecting the existing nodes with the Steiner ones through an arc whose profit will be the revenue of the original node. The binary arc variables $x_a^t \in \{0,1\}$ take value 1 if arc a representing the connection of node i is connected for the first time in time period t, and 0 otherwise. Each arc has a profit p_a^t which corresponds to minus the cost of the arc it represents or the revenue of the node it represents. Constants c_a^t and d_a represent the costs of building arc a in time period t (again, this may also include maintenance costs for the following time periods) and the distance traveled to build it, respectively. Considering $a \in A$, a^* indicates the arc pointing into the opposite direction of a if a^* exists. $P(a)$ designates the index set of only direct predecessor arcs of a, with the exclusion of a^*. Naturally, arcs incident to the root node (artificial or not) have no predecessor. A cycle C is defined in the same way as in Section 3.1: a path of arcs originating and ending at a given node. Z is the solution graph found at each branch-and-bound node. Their formulation is as follows.

$$Max \sum_{t \in T} \Big(\sum_{a \in A} p_a^t \cdot x_a^t \Big). \tag{3.22}$$

Subject to

GSECs

$$\sum_{t \in T} \sum_{a \in C} x_a^t \leq |C| - 1, \ \forall C \in Z. \tag{3.23}$$

Multi-period constraints

$$\sum_{t \in T} x_a^t \leqslant 1, \ \forall a \in A, \forall a^* \notin A. \tag{3.24}$$

Side constraints

$$\sum_{a \in A} c_a^t \cdot x_a^t \leqslant budgetLimit^t, \ \forall t \in T. \tag{3.25}$$

$$\sum_{a \in A} d_a \cdot x_a^t \leqslant distanceLimit^t, \ \forall t \in T \tag{3.26}$$

Connectivity constraints

$$x_a^t \leqslant \sum_{t'=1}^{t} \sum_{b \in P(a)} x_b^{t'}, \ \forall a \in A, P(a) \neq \varnothing, t \in T. \tag{3.27}$$

Arc covering constraints

$$\sum_{t \in T} x_a^t + \sum_{t \in T} x_{a^*}^t \leqslant 1, \ \forall a, a^* \in A. \tag{3.28}$$

Variable domains

$$x_a^t \in \{0, 1\}, \forall a \in A, t \in T. \tag{3.29}$$

Constraints (3.23) are exponentially many and guarantee the exclusion of cycles within a given time period. They were thoroughly explained in Section 3.1. Inequalities (3.24) ensure that each arc can be selected at most once throughout the planning horizon. Constraints (3.25) limit the amount of arcs in period t by expressing a maximum budget requirement for each time period. Constraints (3.26) work in a similar way, limiting the amount of arcs built in time period t by a maximum distance requirement for each time period. The connectivity constraints (3.27) guarantee that an arc may only be covered in period t, if at least one predecessor arc is covered in periods $t' \leqslant t$. The arc covering constraints (3.28) guarantee that either arc a or arc a^* (reverse direction) may enter the network in the planning horizon.

The authors have replaced one by one the y-variables for x-variables, which means their formulation has the exact same amount of variables than our own. They use the undirected GSEC inequalities to exclude cycles at each time period, which, as discussed in Section 3.1, is a methodology mathematically corroborated by the works of Fischetti (1991); Chopra and Rao (1994a,b); Feremans et al. (2002); Ljubić et al. (2005); Magnanti and Raghavan (2005); Ljubić et al. (2006); Costa et al. (2009) to be weaker than the one we propose. Moreover, even though they use *branch-and-cut* as a solving method and insert violated subtour elimination constraints if the IP solution presents disconnected cycles, they do not mention any fractional separation algorithm to dynamically identify the constraints that have to be added to the model. The instances solved are therefore considerably small in comparison to those that are solved to optimality in our work, as can be seen in Chapter 5. Moreover, our side constraints that consider the budget limit (3.11) are broader than the ones considered in Suhl and Hilbert (1998), described in (3.25), due to the flexibility of choosing subsets that may be of one time period but also of multiple consecutive time periods.

Chapter 4
Branch and Cut Algorithm

4.1
Data Pre-processing

In order to reduce the size of the problem considered, we perform a pre-processing step. It concerns the budget available for specific subsets of time periods and the distance limit available for each time period. The idea is to only consider those arcs that can be connected within the given budget and distance limits for each time period. Those arcs can be efficiently identified running a shortest path algorithm (Cormen et al., 2009) between the nodes that already belong to the network and all candidate nodes that can be added during the studied horizon. The goal is to find out which nodes are within the budget and distance limits and which nodes can be eliminated from the instance because they are further beyond the defined limits.

4.2
Incorporating Cut Constraints

The constraint set (3.8) ensures that, in each time period, all solution network vertices are connected to the root node. It does so by guaranteeing that, for every subset $W \subseteq V$ that includes a vertex i and does not include the root vertex v_0, at least one of the arcs in the set of all incoming arcs in W must be chosen to be in the solution if node i is connected.

Since the cut set is exponentially large, the constraints cannot all be added to the MILP model at the beginning. We therefore *separate* them dynamically during the optimization process. At each node of the branch-and-bound tree, when an integer or fractional infeasible solution is found, the separation algorithms identify cut constraints that are violated by the current solution, and, hence, add them to the model.

4.3
Separation Algorithms

We separate the constraints of type (3.8) during the optimization process using the separation procedures described in 4.3.1 and 4.3.2. Implementing more and more sophisticated separation procedures along with state-of-the-art branching strategies is a typical plan to attack increasingly complicated instances (Fischetti et al., 2017). An efficient separation of violated inequalities is crucial to tackle complex problem instances.

A pseudo-algorithm that illustrates the solving method is described in Algorithm 1. The algorithm terminates after proving optimality, or after reaching the given time limit.

Algorithm 1 *Branch-and-Cut* for the MPCSTB

```
1:  Initialization: Pre-process initial problem and put on Node list
2:  while !(Node list empty) OR !(Achieve time limit) do
3:      Choose and remove a node from Node list
4:      Solve LP relaxation
5:      if (Infeasible) OR (Relaxed solution value ≥ Incumbent solution value) then
6:          Prune
7:      end if
8:      if Integer infeasible solution found then
9:          Call separation algorithm for integer infeasible solutions to find violated inequalities
10:         Add violated inequalities
11:         Go to line 4 (solve new LP relaxation)
12:     end if
13:     if Fractional infeasible solution found then
14:         Call separation algorithm for fractional infeasible solutions to find violated inequalities
15:         Add violated inequalities
16:         Go to line 4 (solve new LP relaxation)
17:     end if
18:     if Node = root node then
19:         Call Primal Heuristic
20:         if Incumbent solution value found < Previous incumbent solution value then
21:             Update incumbent solution
22:         end if
23:     else
24:         Branch: create new problems and add them to Node list
25:     end if
26: end while
```

4.3.1 Integer Infeasible Solutions

Our *branch-and-cut* approach includes cutting off infeasible integer points as well as infeasible fractional ones. In this section, we describe the algorithm used to separate the integer infeasible solutions. Such solutions may have been enumerated during the branching procedure or may even have been detected by the heuristics of the MIP solver, since the set of constraints (3.8) is not provided to it and therefore the solver was not given the complete structure of the problem (Fischetti et al., 2017).

Generally speaking, to separate integer solutions, we find the connected components of a selected vertex. If they do not include the root vertex, we insert the cut. The algorithm that separates the integer infeasible solutions has a complexity of $O(n+m)$, where n is the number of vertices and m is the number of arcs of the instance, and works as follows and as shown in Algorithm 2.

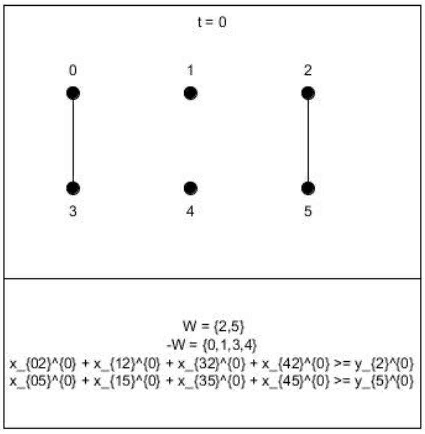
Algorithm 2 Separation procedure at integer nodes

Data: The connected components of the current integer infeasible solution at time period t, found by running BFS.
Result: A set of violated inequalities incorporated into the current LP.
while !(Exist only one connected component including the root node) **do**
  for Each connected component that does not include the root node **do**
    Create set W that does not include the root node and contains the connected component.
    Create set $\overline{W}$, complementary to set W, containing the root and all other vertices.
    for $w \in W$ **do**
      Insert the violated cut $\sum_{t' \leq t} \sum_{v \in \overline{W}} x_{vw}^{t'} \geq \sum_{t' \leq t} y_w^{t'}$ into the LP.
    end for
  end for
end while

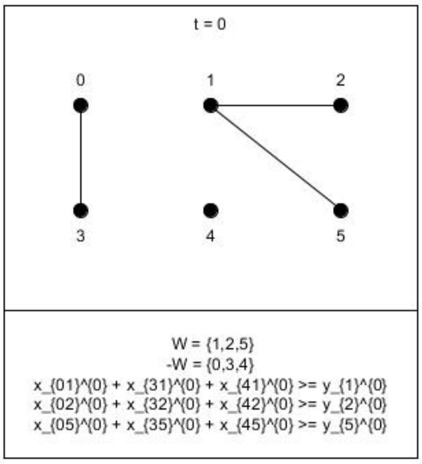
We first compute the connected components of the integer infeasible solution. A connected component is a set of vertices in a graph that are linked to each other by paths. They can be found by running a standard Breadth-First Search (BFS) on the original graph. For example, in Figure 4.1 for a 6-vertices graph, two connected components sets are found : $\{0,3\}$ and $\{2,5\}$.

W does not contain the root and contains vertices with $y_i^t > 0$. So, in this example, where the root is represented by $\{0\}$, $W = \{2,5\}$. Its complementary set contains the root and all other vertices: $\overline{W} = \{0,1,3,4\}$. The two sets must contain all the vertices in the problem.

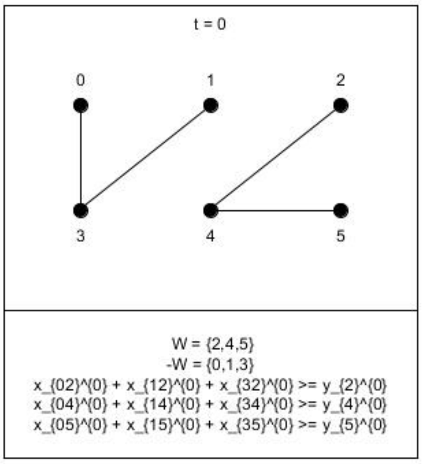
The cuts for that specific integer infeasible solution must be added. Since the network solution must be connected, there must be an arc that goes from set $\overline{W}$ to set W. That obligation is reflected in the cut constraints added to the problem. In our example, those would be (supposing the integer infeasible solution is found in the first time period, represented by 0): $x_{02}^0 + x_{12}^0 + x_{32}^0 + x_{42}^0 \geq y_2^0$ and $x_{05}^0 + x_{15}^0 + x_{35}^0 + x_{45}^0 \geq y_5^0$. This iteration and the following iterations are shown in Figure 4.1.



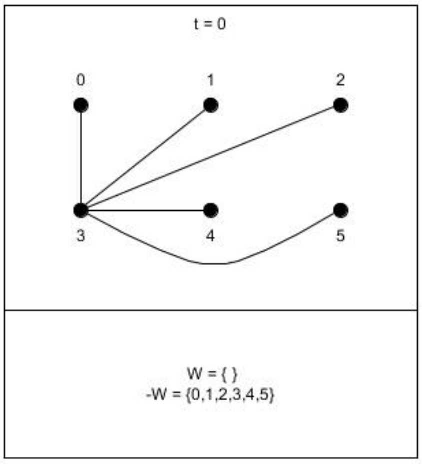
(a) Example of an integer infeasible solution, the sets W and $\overline{W}$, and the cuts that have to be inserted.



(b) The cuts inserted in the previous iteration led to this integer infeasible solution. x_{12}^0 and x_{15}^0 were added to the solution. Sets W and $\overline{W}$ are updated and new cuts are inserted.



(c) Again, the cuts inserted in the previous iterations led to this integer infeasible solution. x_{31}^0, x_{42}^0 and x_{45}^0 were added to the solution. Sets W and $\overline{W}$ are updated and new cuts are inserted.



(d) The cuts inserted in the previous iterations guaranteed an integer feasible solution. The separation procedure ends.

Figure 4.1: Example of separation procedure for an integer infeasible solution.

4.3.2
Fractional Infeasible Solutions

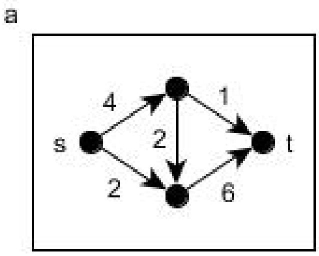
Generally speaking, for fractional solutions, cut constraints are separated by the calculation of the maximum flow value (Fischetti et al., 2017). For $W \subseteq V$, define A_W as the set of arcs with both endpoints in W. An LP-solution $(\hat{x}, \hat{y})$ is found. Then, a support graph $G_W = (W, A_W, \hat{x})$ is built. The arc capacities of such support graph are defined as $\hat{x}^t_{ij}$ for all $(i, j, t) \in A_W$. Subsequently, the maximum flow is calculated from the root node v_0 to each vertex $i \in W$ that has $\hat{y}^t_i > 0$. A violated inequality is inserted into the LP for each maximum flow value found that is less than $\hat{y}^t_i$. Such violated inequality is induced by the corresponding min-cut in the graph G_W (Gollowitzer and Ljubić, 2011).

It is important to introduce the Maximum flow problem, as well as the Minimum cut problem and the Max-flow Min-cut theorem to fully understand the core of our solving method.

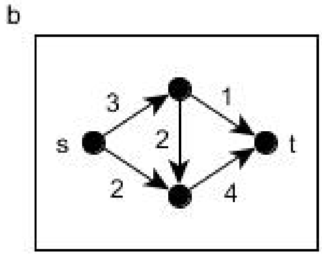
4.3.2.1
Maximum flow and Minimum cut problems

A clear intuition for the Maximum flow problem is to imagine a hydraulic network. Consider a series of pipelines through which a fluid flows from a point of origin s, named source, to a point of destination k, named sink. This network is represented by a graph G. The Maximum flow problem is simply described as finding out the maximum flow that can travel from source to sink, given the capacities of the pipelines in the network. Another straightforward analysis for a network is of a railway that transports commodities. We are interested to know not only which is the maximum quantity of products that can be transported during one day, but also which is the minimum number of railway tracks that, if malfunctioned, would stop completely the flow of products from source to sink. This is defined as the Minimum cut problem. When mathematical programming is concerned, as can be seen in Conforti et al. (2014), these problems are dual problems. That is, if a solution is found for one of these problems, an equivalent solution is simultaneously found for the other one.

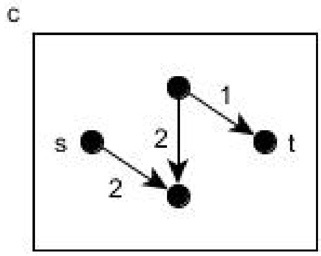
Formally, the problems are presented in the following way. Consider a directed graph $G = (V, A)$ and the function $cap : (V, V) \to \mathbb{R}^+$ that attributes capacities to each arc a. $cap(u, v)$ equals zero if (u, v) is not an arc in graph G. Consider as well two nodes s and k of V, source and sink, respectively. Additionally, a function $flow : (V, V) \to \mathbb{R}^+$ represents a flow in graph G given that the following constraints are respected:



(a) Network with capacities



(b) Maximum flow



(c) Minimum cut

Figure 4.2: Example of Max-flow and Min-cut from Conforti et al. (2014).

Capacity constraints

$$flow(u,v) \leqslant cap(u,v),\ \forall (u,v) \in A. \tag{4.1}$$

Anti-symmetry constraints

$$flow(u,v) = -flow(u,v),\ \forall (u,v) \in A. \tag{4.2}$$

Conservation of flow

$$\sum_{v \in V} flow(u,v) = 0,\ \forall u \in V \setminus \{s,k\}. \tag{4.3}$$

The *Maximum flow problem* consists in finding a feasible s,k-flow of maximum value that respects the constraints (4.1), (4.2) and (4.3). See Figure 4.2, originally from Conforti et al. (2014), for an example. The Maximum flow problem's objective function can be formulated as:

$$Max \sum_{v \in V} flow(s,v).$$

A closely related problem is the *Minimum cut problem.* An s,k*-cut* in the network is a bipartition of the set of vertices V given that s is in S and k is in K. That is, an s,k*-cut* is a division of the vertices of the network into two parts, with the source in one part and the sink in the other. The cut capacity is given by the expression:

$$cap(S, K) = \sum_{u \in S} \sum_{v \in K} cap(u, v). \tag{4.4}$$

Thus, if all the arcs in the cut-set found are removed, then no positive flow is possible, because there is no path in the resulting graph from the source to the sink. The Minimum cut problem consists in finding an s, k-cut of minimum capacity:

$$Min\ cap(S, K).$$

A classical theorem of Ford and Fulkerson (2009) states that the maximum value of an s, k-flow and the minimum capacity of an s, k-cut coincide. It is called the Max-flow Min-cut theorem and links the maximum flow through a network with the minimum cut of the network, that is, the maximum value of an s-k flow equals the minimum capacity over all s-k cuts. Generally, this means that the maximum amount of flow that can be transferred from the source to the sink equals the total capacity of the arcs that are present in the minimum cut. The Maximum flow problem and the Minimum cut problem can be formulated as two primal-dual linear programs. The Max-flow Min-cut theorem is a special case of the duality theorem for linear programs and it is explained in detail in Conforti et al. (2014).

4.3.2.2
Push-relabel maximum flow algorithm

The push-relabel algorithm is an algorithm for computing maximum flows based on the intuition of a hydraulic network. A number of different levels are defined. The source is initially at the highest level whilst all other vertices are at the lowest level. The fluid flows always from a highest to a lowest level and, at each iteration, the algorithm tries to move up the level of a vertex (operation named *relabel*) and push more flow through the network (operation named *push*). In comparison, the Ford-Fulkerson algorithm (Ford and Fulkerson, 2009) is based on the idea of augmenting paths. At each iteration, the algorithm searches a path to pass flow and performs these global augmentations that intend to send flow from the source to the sink. The push-relabel approach is more efficient than the Ford-Fulkerson algorithm. While the time complexity for Ford-Fulkerson is $O(f * m)$, where f is the maximum flow value and m is the number of arcs of the graph, the push-relabel maximum flow algorithm (Cherkassky and Goldberg, 1995) has a strongly polynomial

$O(n^2m)$ time complexity (Cormen et al., 2009).

For finding the maximum flow in a directed graph, we used an adaptation of Goldberg's push-relabel maximum flow algorithm (Cherkassky and Goldberg, 1995). Variables are: (a) a current flow $flow(u, v)$ for each arc $(u, v) \in W$; (b) a capacity $cap(u, v)$, also associated to each arc (u, v); (c) a height $h(u)$, that is used to measure if a vertex u can push flow to an adjacent node v, as the operation can only happen to a smaller height vertex; (d) an excess flow $e(u)$ associated to each vertex u, which is the flow balance in a vertex, i.e., the difference between the total flow coming into the vertex and the total flow going out of the vertex. The two main operations in the algorithm are:

- $Push(u, v)$: push as much flow as possible from u to v, as shown in the pseudo-algorithm 3.
- $Relabel(u)$: relabel u as much as possible without violating the height constraint on the node, as shown in the pseudo-algorithm 4.

Algorithm 3 $Push(u, v)$:

$\delta := min\{e(u), cap(u, v)\}$
$flow(u, v) := flow(u, v) + \delta$
$cap(u, v) := cap(u, v) - \delta$
$cap(v, u) := cap(v, u) + \delta$
$e(u) := e(u) - \delta$
$e(v) := e(v) + \delta$

Algorithm 4 $Relabel(u)$:

$h(u) := min\{h(v) + 1 | (u, v) \in W\}$

4.3.2.3 Separation procedure algorithm

The outline of the separation procedure is given in Algorithm 5. Ljubić et al. (2005) originally presented Algorithm 5 for a single time period. The separation algorithm is executed independently for every time period of the study horizon.

The input of the algorithm is a support graph of the form $G_W = (W, A_W, \hat{x})$ that, as specified in Section 4.3.2, is built from the set of vertices $W \subseteq V$, the set of arcs A_W and the relaxed solution $(\hat{x}, \hat{y})$. We compute the maximum flow on the support graph for all (v_0, i) pairs of vertices, where

Algorithm 5 Separation procedure at fractional nodes

Data: A support graph $G_W = (W, A_W, \hat{x})$.
Result: A set of violated inequalities incorporated into the current LP.
for $i \in W \mid \hat{y}_i^t > 0$ **do**
  $f = MaxFlow(G_W, \hat{x}, v_0, i, W_{v_0}, W_i)$;
  Detect the cut $\delta^+(W_{v_0})$ such that $\hat{x}(\delta^+(W_{v_0})) = f, v_0 \in W_{v_0}$;
  if $f < \hat{y}_i^t$ **then**
    Insert the violated cut $x(\delta^+(W_{v_0})) \geq y_i^t$ into the LP;
  end if
end for

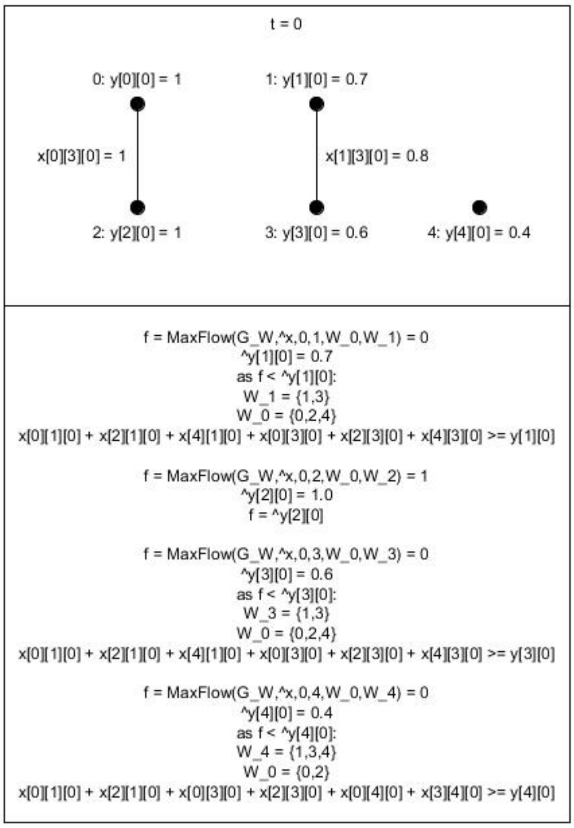
$i \in W$ and $\hat{y}_i^t > 0$. Goldberg's implementation (Cherkassky and Goldberg, 1995) of the push-relabel maximum flow algorithm returns in one calculation not only the maximum flow value $f = MaxFlow(G_W, \hat{x}, v_0, i, W_{v_0}, W_i)$ but also both sets $W_{v_0}, v_0 \in W_{v_0}$ and $W_i, i \in W_i$ that together define the minimum cut of value f (Ljubić et al., 2005). Subset $W_{v_0} \subset W$ contains root vertex v_0 and induces a minimum cut closest to v_0, in other words, as established by the Max-flow Min-cut theorem (Conforti et al., 2014), $x(\delta^+(W_{v_0})) = f$. At the same time, subset $W_i \subset W$ contains vertex i and induces a minimum cut closest to i, i.e., $x(\delta^-(W_i)) = f$. Finally, if $f < \hat{y}_i^t$, we insert the violated cut $x(\delta^+(W_{v_0})) \geq y_i^t$ into the LP. An example of the separation procedure is shown in Figure 4.3.

4.4 Primal Heuristic

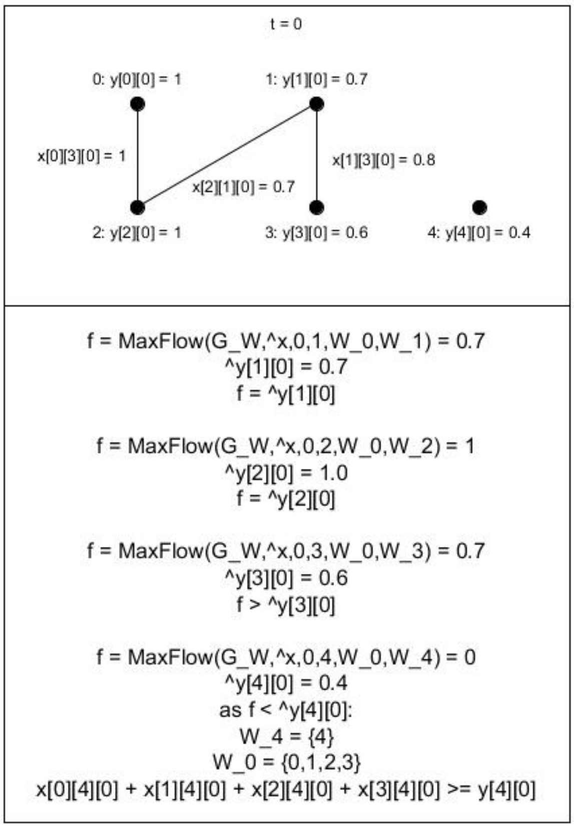
We have developed a primal heuristic to improve the upper bound of the problem. That is, the best integer feasible solution the model is able to find. Our heuristic is called only in the root node of the branch-and-bound tree, before a branch is performed, once the linear program is solved and no more violated inequalities are found. (Ljubić et al., 2005).

The general idea of our algorithm is to pick the most promising vertices for our heuristic solution and, through Kruskal's minimum spanning tree heuristic (Cormen et al., 2009), choose the most promising edges connecting these vertices. After that, respecting the budget and distance constraints, decide on which period of the study horizon each of the selected vertices and arcs are built.

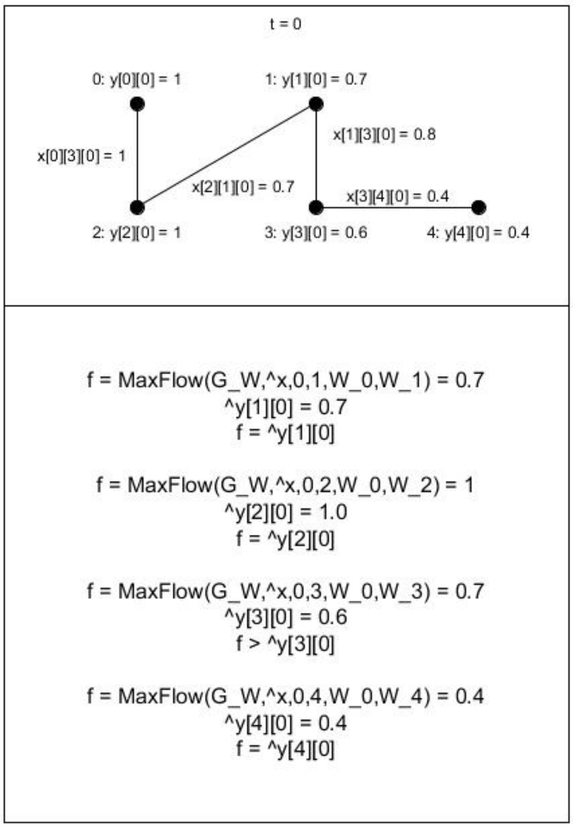
The first step taken by the algorithm is the selection of a set of vertices S from graph $G = (V, A, c)$ that will be the core of the heuristic solution. We aim to build the strongest heuristic solution possible. Therefore, to select the most promising vertices to be in set S, we use the information of the fractional values of the y-variables in the LP-solution of the current node in the *branch-*



(a) Example of a fractional infeasible solution, the max-flow value between source and sink and the cuts that have to be inserted.



(b) The cuts inserted in the previous iteration led to this fractional infeasible solution. New max-flow values are calculated and new cuts are inserted.



(c) The cuts inserted in the previous iterations guaranteed a fractional feasible solution. The separation procedure ends.

Figure 4.3: Example of separation procedure for a fractional infeasible solution.

and-cut tree. For each vertex, we sum the y-values for every period of the study horizon and check if the sum is greater than 0.5. If it is, that vertex is selected to set S.

Next, the distance network G_S is calculated for S, where $G_S = (S, S \times S, d_S)$. We define the length d_S of an edge in G_S as the length of the shortest path connecting the two corresponding vertices in G. The shortest path matrix is calculated using the Floyd-Warshall algorithm (Cormen et al., 2009). In the interest of choosing the best paths between vertices of set S, we determine the length of an edge as the value 1 minus the solution value of that edge. The solution value of an edge is the maximum value between the solution values of the x-variables in the LP-solution in the two arcs that define that edge. Mathematically, we assign to each edge (i, j, t) the cost $(1 - max\{\hat{x}^t_{ij}, \hat{x}^t_{ji}\})$ where $\hat{x}^t_{ij}$ is the value of the corresponding x-variable in the fractional solution of the current branch-and-bound node. After all, the shortest path is the path with high fractional values of the x-variables in the LP-solution (Ljubić et al., 2005).

Then, we compute Kruskal's minimum spanning tree $Z = (S, A_Z)$ (Cormen et al., 2009) in G_S. Naturally, there are vertices on the shortest paths that correspond to arcs in A_Z. Therefore, we define the set S' of vertices in G as the union of S and the set of all these extra vertices that show up along the shortest paths. Consequently, $G_H = (S', A_H, c)$ is defined as the subgraph of G induced by the vertex set S'. Notice that in G_H, the cost of each arc is again the original cost in the problem instance.

Evidently, G_H is connected, and therefore we are able to compute Kruskal's minimum spanning tree $Z' = (S', A_{Z'})$ (Cormen et al., 2009) for it. This procedure is done to try to find an heuristic solution as tight as possible to serve as an upper bound to our problem. We end up with a single-period solution that we need to manually separate into several time periods.

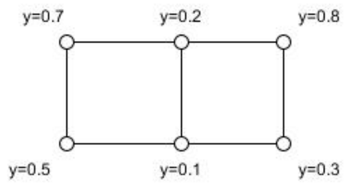
In the effort to separate the solution Z' into time periods as optimally as possible, we use a greedy algorithm based on the fractional solution values of the y-variables for each vertex in Z'. The initial vertex is v_0, no matter if the instance has an actual root node or if that root node is artificially created in the algorithm. Then, comes the node i that is connected to v_0. After that, there may be only one possibility of node to connect to node i or even a multitude of possibilities. That will depend on the format of the single-period solution tree Z'. In the event there are multiple possibilities, the greedy algorithm acts to choose the node connected to vertex i with higher fractional y-solution value. And so on and so forth until either the budget limit or the distance limit are achieved and the time period is increased. Note that if the study horizon ends and there are still nodes in Z' to be inserted in the multi-period Z'' solution, said nodes are discarded due to the budget and distance limits constraints that must be applied.

We therefore attempt to solve the MPCSTB on Z by the linear time algorithm described and shown in Algorithm 6. Naturally our heuristic solution is viable (even if not optimal) because it makes sure that the vertices in S' can be connected as there are paths between them in the input instance, guaranteed by the Floyd-Warshall algorithm. Moreover, they are connected in the best way possible through a minimum spanning tree, that, by definition, connects all vertices in S' together, without any cycles and with the minimum possible total arc weight. Furthermore, we are careful to respect the side constraints, separating the resulting minimum spanning tree Z' by time periods, applying a greedy algorithm. This algorithm, while respecting the budget and limit constraints, chooses to insert in the heuristic solution the maximum number of vertices of Z' possible in the first period, the maximum number of remaining vertices of Z' in the second period and so on and so forth, in the best order possible, as the objective function value decreases by this practice. In sum, all the constraints in the problem are respected, guaranteeing a viable heuristic solution.

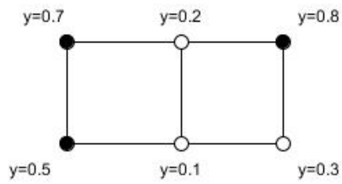
Algorithm 6 Primal Heuristic

Data: Solution of LP relaxation $(\hat{x}, \hat{y})$.
Result: An heuristic solution Z''.
if $\sum_{t \in T} \hat{y}_i^t \geq 0.5$ **then**
    $S \leftarrow S \cup \{i\}$
end if
Calculate $\hat{x}_{ij} = max\{\hat{x}_{ij}^t\} \mid \forall i, j \in S$
Calculate $l_S = (1 - max\{\hat{x}_{ij}, \hat{x}_{ji}\}) \mid \forall i, j \in S$
Calculate $d_S = \text{Floyd-Warshall}(l_S)$
Compute distance network $G_S = (S, S \times S, d_S)$
Compute Kruskal's $Z = (S, A_Z)$ in G_S
Define S_{sp} as the set of all vertices on the shortest paths in A_Z
Define $S' = S \cup S_{sp}$
Define $G_H = (S', A_H, c)$
Compute Kruskal's $Z' = (S', A_{Z'})$
Separate single-period Z' into multi-period Z'' by greedy algorithm

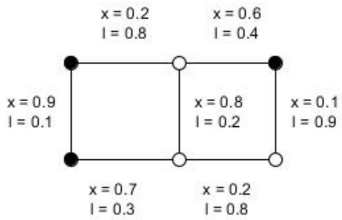
Figure 4.4 shows an example of the primal heuristic procedure. The computational results in Chapter 5 show that this heuristic may significantly facilitate the search for feasible solutions, and therefore the entire solution process, as great improvement can be seen in the gap between the lower bound and the best known feasible solution for our most challenging problem instances.



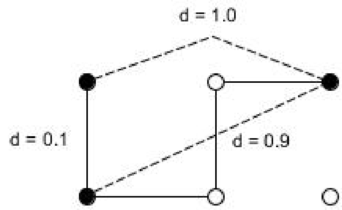
(a) Solution of LP relaxation with vertices and their respective $\hat{y}$.



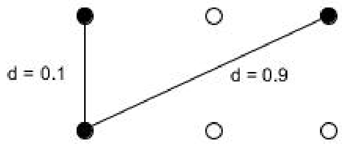
(b) Vertices in set S shown in black.



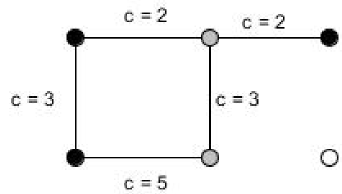
(c) Selected edges and their respective $\hat{x}$ and l_S.



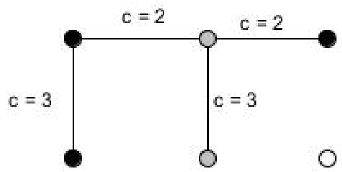
(d) d_S = Floyd-Warshall(l_S) and distance network $G_S = (S, S \times S, d_S)$.



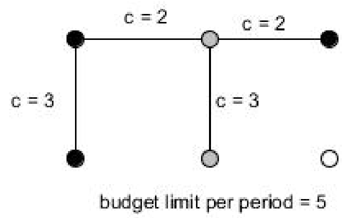
(e) $Z = (S, A_Z)$ in G_S.



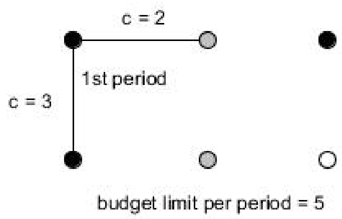
(f) S_{sp} shown in grey and $S' = S \cup S_{sp}$ shown in black and grey and finally $G_H = (S', A_H, c)$.



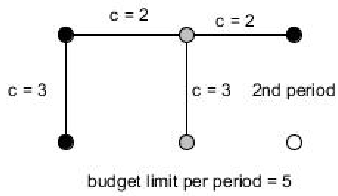
(g) $Z' = (S', A_{Z'})$.



(h) Single period Z' with the budget limit per period information.



(i) First-period Z''.



(j) Multi-period Z'' complete.

Figure 4.4: Example of primal heuristic procedure.

Chapter 5
Computational Results

We have tested the proposed *branch-and-cut* algorithm outlined in this thesis extensively on three different sets of instances:

- In Fischetti et al. (2017), Fischetti et al. tested their exact *branch-and-cut* algorithm on the "PUCNU" dataset, instances based on the PUC series (Rosseti et al., 2003) for the classical Steiner problem in graphs. Given that those instances are designed for one single time period, all PUCNU instances were transformed into multi-period instances with numbers of periods equal to 2, 3, 5 and 8.
- We have randomly generated instances of complete graphs (named "CG instances"), by drawing random points in a defined Cartesian plan. All edges between nodes are present in this set. Their costs are equal to the Euclidean distances between their vertices. Multi-period instances are created with number of periods equal to 2, 3, 5 and 8.
- Finally, we have also randomly generated instances of incomplete graphs (named "IG instances"), by drawing random points in a defined Cartesian plan. Some edges are selected to have edge costs equal to the Euclidean distances between their nodes. Graphs are incomplete, but checked for connectivity. Multi-period instances are created, similarly to the CG dataset, with numbers of periods equal to 2, 3, 5, 8, 10 and 15.

All computational experiments were carried out on a Linux Mint 18.3 Cinnamon 64-bit operating system, 3.6.7 Cinnamon version with 3.40 GHz Intel processor and 16 GB of RAM. The algorithm was programmed in Java programming language and, as mentioned before, to solve the proposed integer linear programming (ILP) formulation, we make use of a *branch-and-cut* algorithm. As a generic implementation of the *branch-and-cut* approach, we used the commercial packages ILOG CPLEX and ILOG Concert Technology, version 12.7.1. (IBM, 2017). This means we solve the ILP, initially without cut constraints (3.8), using the MIP solver from CPLEX and then we call user-callbacks to add cut constraints on the fly. At each node of the branch-and-bound tree we solve the LP-relaxation, obtained by replacing the integrality requirements (3.14) and (3.15) by the simple bounds: $0 \leq x_{ij}^t \leq 1, \forall (i,j) \in$

$A, \forall t \in T$ and $0 \leq y_i^t \leq 1, \forall i \in V \backslash \{v_0\}, \forall t \in T$. Integer infeasible points are cut off by means of a **LazyConstraintCallback** where we insert our separation algorithm, explained in Section 4.3.1, while fractional infeasible points are cut off by means of a **UserCutCallback**, that contains the separation procedure explained in Section 4.3.2. The **UserCutCallback** will be used within the cut loop that CPLEX calls at each node of the *branch-and-cut* algorithm. It will be called after CPLEX has ended its own cut generation loop so that we can specify additional cuts to be added to the cut pool. Moreover, the *branch-and-cut* framework of CPLEX calls an **HeuristicCallback** where we implemented our own primal heuristic, detailed in Section 4.4.

5.1 Distance and budget limits estimation

The PUCNU dataset was altered to provide multi-period instances for the MPCSTB problem and the CG and IG datasets were randomly generated. Therefore, all sets of instances make use of the artificial root constraints (3.18), (3.19) and (3.20) and compel us to calculate distance and budget limits for these three sets of instances that would challenge the algorithm into finding the optimal solution. Hence, we considered as a good estimation of a tight distance limit per period for our set of instances the total average distance built of edges added throughout the whole time horizon divided by the number of periods in the time horizon, given in Equation 5.1.

$$distanceLimit^t = \frac{\overline{d}}{|T|}, \ \forall t \in T \tag{5.1}$$

$$\overline{d} = D_a \times T_n \tag{5.2}$$

Equation 5.2 shows that a good estimation of the total average distance built $\overline{d}$ spent in the whole time horizon could be the average distance D_a of building an edge times the number of terminals T_n that would maximize the number of ways to combine k terminals from a set of nT terminals. Afterall, we are interested in allowing a maximum number of terminals to enter the network in our study horizon. A k-combination of a set of terminals T_L is a subset of k distinct elements of T_L. If the set has nT elements, the number of k-combinations is equal to the binomial coefficient, as can be seen in Equation 5.3.

$$\binom{nT}{k} = \frac{nT!}{k!(nT-k)!} \tag{5.3}$$

The k that maximizes the number of ways to combine k terminals from a set of nT terminals is $nT/2$. Hence, the distance limit is calculated for each time period as the number of terminals (nT) times the average distance D_a divided by 2 times the number of total periods, as can be seen in Equation 5.4.

$$distanceLimit^t = \frac{nT \times D_a}{(2 \times |T|)}, \ \forall t \in T \tag{5.4}$$

A tight budget limit is estimated in the same way, but considering the number of time periods in the subset of time periods defined for it, aka, $|\hat{T}_B|$. C_a is the average cost of building an edge. The percentage rate p_r has the purpose of making the budget limit even tighter, as Equation 5.5 shows.

$$budgetLimit^{T_B} = \frac{nT \times C_a \times p_r \times |\hat{T}_B|}{(2 \times |T|)}, \ \forall T_B \in \hat{T}_B \tag{5.5}$$

5.2 Modified PUCNU instances

Rosseti et al. (2003) proposed new test instances named "PUC series" for the Steiner problem in graphs. These instances were meant to be used for the evaluation and comparison of existing and newly developed algorithms. Their cutting-edge characteristics were: (a) non-amenability to reductions proposed by preprocessing techniques; (b) hard to compute lower bounds; (c) large integrality gaps between the optimal integer solution and that of the linear programming relaxation; and (d) symmetry aspects, which made them more difficult to solve to both exact methods and heuristics than the previously existing test instances. The PUC series provided difficulties for well established state-of-the-art heuristics, which used to find optimal solutions for almost all previously used test instances. The latest algorithms found fewer optimal solutions and more discriminant numerical results. Therefore, the PUC series allows a good assessment of the effectiveness and the relative behavior of different exact methods and heuristics.

The PUCNU instances (Fischetti et al., 2017) were based in the PUC series instances. They consider incomplete graphs with edge costs values equal to 1 and vertex profits varying from 0 to 2. A vertex with profit 0 is a Steiner node. A vertex with profit 1 or 2 is a terminal node. The edge costs are the same for all time periods and the profit accumulates: if a vertex enters the network in a certain period, its profit is accounted for at that period and at all following periods. The costs and profits of an instance are so similar, that it offers a lot of possibilities of good solutions, but it is harder for the model to precise which one is optimal. Generally, having similar arc costs and similar profits tends to make finding the optimal solution more difficult. Given that the PUCNU instances were designed for one single time period, they had to be transformed into multi-period instances to test our algorithm.

Table 5.1 informs the characteristics of the PUCNU dataset. For each instance, it summarizes the instance name, the number of vertices, number of edges and the number of terminal nodes (“nT”). Following results will show the best lower ("LB") and upper bound ("UB") found, the gap, the number of nodes searched by the branch-and-bound tree, the number of cuts added by the model and the time it took to run. It is important to point out that a time-limit of one hour was used to run these instances. However, if the time limit is surpassed while a callback is running, the callback is finished before quitting the program. That is why some time markers may have values greater than 3600 seconds. It is important to mention that the results in this Section consider the use of both separation procedures and all connectivity constraints.

Table 5.1: Modified PUCNU instances

name	$\lvert V\rvert$	$\lvert E\rvert$	**nT**
bip42nu	1200	3982	200
bip52nu	2200	7997	200
bip62nu	1200	10002	200
bipa2nu	3300	18073	300
bipe2nu	550	5013	50
cc10-2nu	1024	5120	135
cc11-2nu	2048	11263	244
cc12-2nu	4096	24574	473
cc3-10nu	1000	13500	50
cc3-11nu	1331	19965	61
cc3-12nu	1728	28512	74
cc3-4nu	64	288	8
cc3-5nu	125	750	13
cc5-3nu	243	1215	27
cc6-2nu	64	192	12
cc6-3nu	729	4368	76
cc7-3nu	2187	15308	222
cc9-2nu	512	2304	64

In the interest of full disclosure, we will provide all our PUCNU instances' results in the Appendix A.1. Results for a 2, 3 and 5-periods runs, with all connectivity constraints and both separation procedures are respectively in Tables A.1, A.2 and A.3. In this Section, we will show the results for an 8-

periods run of the PUCNU instances, with two different kinds of settings: one where there is one budget limit for the whole time horizon (Table 5.2) and another where there is a budget limit for time periods 1-4 and a budget limit for time periods 5 to 8 (Table 5.3).

Table 5.2: Modified PUCNU instances | Number of periods: 8 | One budget limit throughout all time horizon

name	**LB**	**UB**	**gap(%)**	**# nodes**	**# cuts**	**time(s)**
bip42nu	1813.01	1831.00	0.98	34	1764	3806.37
bip52nu	1765.67	1781.00	0.86	4	1523	3650.91
bip62nu	1717.40	1733.00	0.90	1	781	3618.07
bipa2nu	-	-	-	-	-	Memout
bipe2nu	373.37	385.00	3.02	20	305	3605.16
cc10-2nu	1206.26	1285.00	6.13	4	2833	3612.95
cc11-2nu	2249.05	2354.00	4.46	1	606	3645.99
cc12-2nu	-	-	-	-	-	Memout
cc3-10nu	367.81	417.00	11.80	6	1504	3614.96
cc3-11nu	573.67	618.00	7.17	3	827	3624.34
cc3-12nu	685.50	746.00	8.11	1	361	3639.49
cc3-4nu	51.99	65.00	20.01	21	1097	3671.80
cc3-5nu	111.27	118.00	5.70	113	2180	3705.38
cc5-3nu	232.20	257.00	9.65	16	2331	3894.27
cc6-2nu	88.59	99.00	10.51	21	1022	3707.83
cc6-3nu	684.50	730.00	6.23	7	2419	3607.58
cc7-3nu	2059.48	2178.00	5.44	1	1053	3653.51
cc9-2nu	580.28	603.00	3.77	14	5506	3603.97

Table 5.3: Modified PUCNU instances | Number of periods: 8 | A budget limit for 1-4 and another for 5-8

name	**LB**	**UB**	**gap(%)**	**# nodes**	**# cuts**	**time(s)**
bip42nu	1879.99	1884.00	0.21	99	3101	3630.98
bip52nu	1833.40	1850.00	0.90	2	810	3651.62
bip62nu	1790.91	1822.00	1.71	3	820	3605.00
bipa2nu	-	-	-	-	-	Memout
bipe2nu	390.73	402.00	2.80	7	738	3605.25
cc10-2nu	1245.15	1323.00	5.88	6	4091	3612.79

Continued on next page

Table 5.3 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
cc11-2nu	2324.57	2423.00	4.06	0	910	3646.51
cc12-2nu	-	-	-	-	-	Memout
cc3-10nu	381.10	434.00	12.19	5	1324	3615.03
cc3-11nu	588.27	640.00	8.08	3	859	3623.98
cc3-12nu	711.00	744.00	4.44	2	608	3639.60
cc3-4nu	55.37	64.00	13.48	23	1042	3686.08
cc3-5nu	115.54	122.00	5.29	53	1462	3614.99
cc5-3nu	239.53	262.00	8.58	25	2508	3654.69
cc6-2nu	92.03	102.00	9.78	29	1259	3603.03
cc6-3nu	708.01	758.00	6.59	10	3049	3607.21
cc7-3nu	2124.20	2244.00	5.34	1	1461	3653.22
cc9-2nu	592.74	615.00	3.62	9	3364	3754.10

As can be seen in Appendix A.1, the larger instance **cc12-2nu** can no longer be solved for a number of periods of 5 or greater due to lack of memory. This means that the CPLEX solver reached its memory limit and could not build the model of the size intended. The same happens for the second largest instance **bipa2nu** for an 8-periods run.

It is important to notice that some instances are solved in the root node of the branch-and-bound tree (every time **# nodes** equals 0). The gaps observed for both 8-periods runs are similar in magnitude, no matter if the budget is limited for the whole horizon or for periods 1-4 and 5-8. To analyze the impact of the different side constraints, we can compare the structure of the best integer feasible solutions found (optimal or otherwise) for each setting. This information is shown in Table 5.4 for the case there is one budget limit throughout all time horizon and in Table 5.5 for the case there is one budget limit for time periods 1 to 4 and another for time periods 5-8. The tables show the number of terminals that are present in the integer feasible solution found, the budget limit value and distance limit value calculated for those instances, the total revenue obtained by those instances' solutions and the total amount spent. It can be seen that a budget limit throughout all time horizon allows the terminal vertices to enter the network at an earliest time period than they would for a budget limit per subset of time periods. That conclusion is drawn due to the value of the total revenue obtained for the different integer feasible solutions. Because of a rounding procedure at the calculation of the budget limit, the budget limit may be one unit bigger for the case where there is one budget limit for a subset of time periods. Hence, extra terminals are allowed to enter the network. Even so, the integer feasible solution found has a lower net worth than the net worth for the case where there is only one budget limit per time horizon. That being said, it is important to point out that the subset flexibility feature is crucial to model real case scenarios.

Table 5.4: Modified PUCNU instances | Number of periods: 8 | One budget limit throughout all time horizon | Solution Structure

name	termInSol	budgetLimit	distLimit	totalRev	totalSpent
bip42nu	61	73.00	13.00	690.00	73.00
bip52nu	62	73.00	13.00	708.00	73.00
bip62nu	67	73.00	13.00	756.00	73.00
bipa2nu	1	107.00	19.00	16.00	0.00
bipe2nu	21	23.00	4.00	238.00	23.00
cc10-2nu	34	51.00	9.00	366.00	51.00
cc11-2nu	63	90.00	16.00	688.00	90.00
cc12-2nu	1	168.00	30.00	16.00	0.00
cc3-10nu	13	23.00	4.00	150.00	23.00
cc3-11nu	14	23.00	4.00	164.00	22.00
cc3-12nu	17	28.00	5.00	194.00	28.00
cc3-4nu	3	6.00	1.00	42.00	3.00
cc3-5nu	5	6.00	1.00	56.00	6.00
cc5-3nu	6	12.00	2.00	74.00	11.00
cc6-2nu	4	6.00	1.00	42.00	5.00
cc6-3nu	21	28.00	5.00	210.00	28.00
cc7-3nu	55	79.00	14.00	597.00	79.00
cc9-2nu	15	23.00	4.00	164.00	23.00

Table 5.5: Modified PUCNU instances | Number of periods: 8 | A budget limit for 1-4 and another for 5-8 | Solution Structure

name	termInSol	budgetLimit	distLimit	totalRev	totalSpent
bip42nu	62	37.00	13.00	638.00	74.00
bip52nu	63	37.00	13.00	640.00	74.00
bip62nu	67	37.00	13.00	668.00	74.00
bipa2nu	1	54.00	19.00	16.00	0.00
bipe2nu	22	12.00	4.00	222.00	24.00
cc10-2nu	32	26.00	9.00	323.00	46.00

Continued on next page

Table 5.5 – *Continued from previous page*

name	termInSol	budgetLimit	distLimit	totalRev	totalSpent
cc11-2nu	63	45.00	16.00	619.00	90.00
cc12-2nu	1	84.00	30.00	16.00	0.00
cc3-10nu	13	12.00	4.00	134.00	24.00
cc3-11nu	14	12.00	4.00	144.00	24.00
cc3-12nu	19	14.00	5.00	196.00	28.00
cc3-4nu	4	3.00	1.00	46.00	6.00
cc3-5nu	5	3.00	1.00	52.00	6.00
cc5-3nu	5	6.00	2.00	66.00	8.00
cc6-2nu	4	3.00	1.00	39.00	5.00
cc6-3nu	20	14.00	5.00	182.00	28.00
cc7-3nu	57	40.00	14.00	532.00	80.00
cc9-2nu	16	12.00	4.00	153.00	24.00

An important comparison to show is between our model and Suhl and Hilbert's. Tables 5.6 and 5.7 show these results for a 5-periods run. The results for 1-period (Tables A.12 and A.13), 2-periods (Tables A.14 and A.15) and 3-periods runs (Tables A.16 and A.17) are in Appendix A.1. As our problem is a minimization problem and Suhl and Hilbert's is a maximization problem, lower bounds and upper bounds differ in value and in meaning. For a minimization problem, the upper bound is the best integer feasible solution found whereas the lower bound is the relaxed solution. For a maximization problem, the upper bound is the relaxed solution and the lower bound is the incumbent solution. To facilitate the comparison between the models, we have transformed through Proof 2 the lower and upper bounds of Suhl and Hilbert's model as if their objective function was of the *Goemans and Williamson Minimization* Problem. It can be seen that Suhl and Hilbert's model may have difficulties finding good integer feasible solutions. Our model outperforms Suhl and Hilbert's for most instances, minus for **c3-5nu**, **c5-3nu** and **c6-2nu**.

Table 5.6: Modified PUCNU instances | Number of periods: 5 | One budget limit per period | MPCSTB

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	1237.60	1240.00	0.19	20	1900	3741.92
bip52nu	1209.43	1217.00	0.62	34	1665	3662.16
bip62nu	1186.12	1200.00	1.16	10	1203	3611.56

Continued on next page

Table 5.6 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bipa2nu	1726.99	1763.00	2.04	0	0	3681.66
bipe2nu	276.34	282.00	2.01	207	1617	3630.32
cc10-2nu	815.89	849.00	3.90	5	3464	3608.50
cc11-2nu	1509.45	1567.00	3.67	2	2029	3630.05
cc12-2nu	-	-	-	-	-	Memout
cc3-10nu	264.39	284.00	6.91	9	1888	3609.96
cc3-11nu	377.78	410.00	7.86	6	1336	3615.97
cc3-12nu	455.00	480.00	5.21	5	1014	3625.99
cc3-4nu	44.00	44.00	0.00	27	862	3352.47
cc3-5nu	63.48	70.00	9.32	31	1830	3627.00
cc5-3nu	148.45	159.00	6.63	25	3242	3601.30
cc6-2nu	51.67	58.00	10.91	30	1260	3651.13
cc6-3nu	454.00	486.00	6.58	9	3173	3605.35
cc7-3nu	1359.74	1427.00	4.71	2	1225	3635.84
cc9-2nu	380.46	392.00	2.94	6	2351	3603.07

Table 5.7: Modified PUCNU instances | Number of periods: 5 | One budget limit per period | SUHL AND HILBERT

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	1205.63	1252.00	3.70	5501	6	3603.57
bip52nu	1182.78	1232.00	4.00	718	0	3600.47
bip62nu	1167.23	1212.00	3.69	398	3	3635.43
bipa2nu	1693.39	2180.00	22.32	0	1	3772.37
bipe2nu	269.93	285.00	5.29	1674	1	3601.10
cc10-2nu	778.85	851.00	8.48	23918	65	3600.37
cc11-2nu	1443.74	1706.00	15.37	6300	34	3603.86
cc12-2nu	-	-	-	-	-	Memout
cc3-10nu	254.29	287.00	11.40	6436	45	3600.16
cc3-11nu	361.43	434.00	16.72	2100	12	3612.27
cc3-12nu	436.81	476.00	8.23	801	3	3816.92
cc3-4nu	44.00	44.00	0.00	221	0	1.55
cc3-5nu	69.00	69.00	0.00	9731	0	61.88
cc5-3nu	148.57	157.00	5.37	200901	44	3600.86
cc6-2nu	55.00	55.00	0.00	4320	0	10.96
cc6-3nu	438.60	481.00	8.81	40801	86	3602.24

Continued on next page

Table 5.7 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
cc7-3nu	1306.80	-	-	6679	51	3600.35
cc9-2nu	366.92	392.00	6.40	98784	26	3600.10

Figure 5.1 shows the best integer feasible solution found by our model for a 5-periods run of instance **cc6-2nu**. We can see the nodes connected by a thick solid line in the first time period, by a dashed line in the second time period, by a dotted line in the third time period, by a dash-dot line in the fourth time period and finally by a thin solid line in the fifth time period. Lower bound, upper bound, gap, number of nodes and cuts and the time it took to run are displayed in Table 5.6.

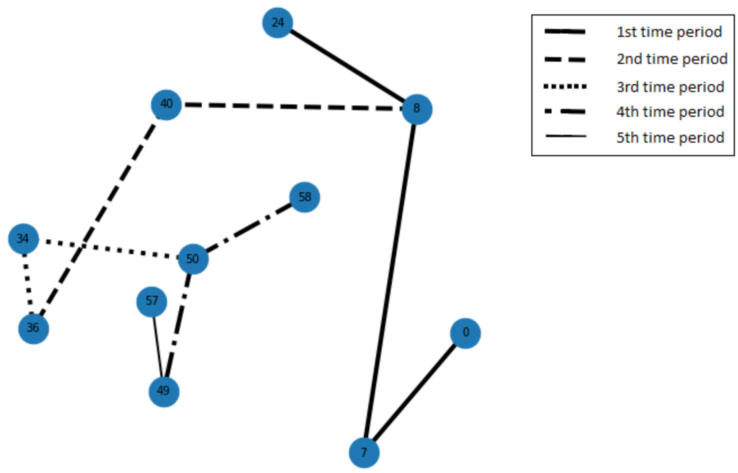


Figure 5.1: Integer feasible solution found for instance **cc6-2nu** for a 5-periods run.

The valid inequalities (3.16) and (3.17) substantially strengthen the model. To confirm their value, we have run the sets of instances for 2 and 3 periods without Equations (3.16) and (3.17). We kept Equation (3.13), as it is mandatory to guarantee the connectivity of the solution. The results are shown in Appendix A.1. When we compare Table A.1 that carries all connectivity constraints and Table A.4 that does not, we can clearly see that all instances' optimality gaps are tighter for Table A.1. When we compare Table A.2 that carries all connectivity constraints and Table A.5 that does not, we see much better results as well for Table A.2.

Another analysis worth completing is the result comparison between using both separation procedures, only the one that separates integer infeasible solutions or only the one that separates fractional infeasible solutions. The complete results are shown in Tables A.6, A.7, A.9 and A.10 in Appendix A.1.

To further analyze these results, we built the summary tables: Table A.8 and Table A.11, also in Appendix A.1, with the best results found for each instance of this dataset, for a 2-periods run and a 3-periods run, respectively. Table 5.8 summarizes these results. We can see that using both separation procedures or only the separation of integer infeasible solutions have shown better results for this particular set of instances. Comparing Tables A.1, A.6 and A.7 and Tables A.2, A.9 and A.10, it can be seen that the results are very similar. Therefore, the conclusion drawn is that the choice of using both separation procedures or just one of them may present tighter gaps depending on the instance run.

Table 5.8: Modified PUCNU instances | Best results

Sep.	**2 periods(%)**	**3 periods(%)**
Int	55.56	61.11
Both	38.89	38.89
Frac	5.56	0.00

5.3 Randomly generated instances of complete graphs

These are randomly generated complete graph instances that aim to emulate real Brazilian gas network expansion instances that cannot be disclosed. The so called “CG instances" are larger than the IG instances as they allow all edges between all nodes in the network to be selected by the model. Therefore, they have the most variables and can test the scalability of our *branch-and-cut* algorithm. Nodes were drawn from a Cartesian plan. Edge costs are the Euclidean distances between nodes. Terminal nodes are randomly selected and constitute approximately 10% of the nodes in each instance while the remaining 90% of nodes are Steiner nodes. Profits of terminal nodes are randomly generated in the same order of magnitude of the edge costs. Again, we have similar costs and profits with the goal of creating instances that provide difficulty to the model in precising which is the optimal solution.

We have generated 5 instances of the same size, to evaluate properly the model’s performance. For each instance, Table 5.9 displays the instance name, the number of vertices, number of edges and the number of terminal nodes (“nT”). Full results are presented in Appendix A.2. Table A.18 presents results for a 2-periods run, Table A.19 shows results for a 5-periods run and Table A.20, for an 8-periods run. In this Section, we will compare the use of different kinds of settings for a 3-periods run: one where there is one budget limit for the whole horizon and another where there is a budget limit for each time period. The results that follow consider the use of both separation procedures and all connectivity constraints. We will also compare our results to Suhl and Hilbert's.

Table 5.9: Randomly generated instances of complete graphs

name	$\lvert V \rvert$	$\lvert E \rvert$	**nT**
50_1	50	1225	4
50_2	50	1225	5
50_3	50	1225	2
50_4	50	1225	3
50_5	50	1225	8
100_1	100	4950	15
100_2	100	4950	14
100_3	100	4950	11
100_4	100	4950	12
100_5	100	4950	10
250_1	250	31125	25
250_2	250	31125	27
250_3	250	31125	19
250_4	250	31125	33
250_5	250	31125	19
500_1	500	124750	44
500_2	500	124750	44
500_3	500	124750	30
500_4	500	124750	53
500_5	500	124750	46
750_1	750	280875	71
750_2	750	280875	72
750_3	750	280875	79
750_4	750	280875	79
750_5	750	280875	66

Table 5.10: CG instances| Number of periods: 3 | One budget limit throughout all time horizon

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	1351.17	1351.17	0.00	23	90	1.77
50_2	1910.28	1910.28	0.00	3	54	1.29
50_3	583.62	583.62	0.00	0	0	0.69
50_4	1139.70	1139.70	0.00	55	301	72.16
50_5	2876.41	2876.41	0.00	228	1871	2713.97
100_1	2406.20	2577.33	6.64	62	1988	3665.06
100_2	2303.27	2967.61	22.39	39	1616	3601.38
100_3	1900.10	2109.53	9.93	64	1764	3601.37
100_4	2212.83	2420.89	8.59	66	2105	3602.43
100_5	1861.48	1861.52	0.00	119	848	670.15
250_1	1947.22	2150.98	9.47	17	1592	3646.23
250_2	1986.52	2062.33	3.68	17	1752	3709.25
250_3	1197.58	1273.62	5.97	72	2484	3733.83
250_4	2129.97	3144.46	32.26	14	1601	3620.48
250_5	1346.87	1627.13	17.22	13	1961	3615.87
500_1	1719.10	1875.06	8.32	6	1484	3713.57
500_2	1546.71	1597.15	3.16	19	2188	3600.73
500_3	1143.46	1200.46	4.75	23	3858	3761.12
500_4	2010.45	2331.80	13.78	9	1797	3611.18
500_5	1571.18	1614.93	2.71	22	1832	3776.70
750_1	1707.24	2090.23	18.32	9	1655	3872.27
750_2	1779.08	2066.99	13.93	7	2017	3883.65
750_3	1918.00	2140.98	10.42	9	2600	3632.77
750_4	1819.45	2049.52	11.23	8	1967	3692.40
750_5	1602.25	1859.45	13.83	12	2384	3631.54

Table 5.11: CG instances | Number of periods: 3 | A budget limit for each time period

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	1351.17	1351.17	0.00	19	207	12.09
50_2	1910.28	1910.28	0.00	3	70	1.25
50_3	583.62	583.62	0.00	0	0	0.69
50_4	1139.70	1139.70	0.00	55	262	27.40

Continued on next page

Table 5.11 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_5	2876.41	2876.41	0.00	253	1977	3246.38
100_1	2422.18	2594.66	6.65	78	2048	3628.73
100_2	2332.64	2552.44	8.61	78	2268	3638.86
100_3	1888.53	2113.52	10.65	40	1613	3739.16
100_4	2210.40	2404.77	8.08	96	2062	3646.94
100_5	1861.52	1861.52	0.00	116	831	755.19
250_1	1937.16	2112.61	8.30	20	2005	3792.39
250_2	1972.25	2069.41	4.70	16	1630	3625.71
250_3	1199.55	1283.05	6.51	27	1850	3657.86
250_4	2131.85	2799.40	23.85	20	2031	3775.56
250_5	1350.60	1627.13	16.99	19	2364	3624.41
500_1	1722.69	1875.06	8.13	7	1524	3645.70
500_2	1546.50	1751.88	11.72	6	1114	3777.97
500_3	1144.24	1200.46	4.68	28	4185	3612.66
500_4	2029.63	2351.09	13.67	10	1852	3667.81
500_5	1571.18	1890.50	16.89	15	1947	3601.58
750_1	1709.10	2090.23	18.23	9	1857	3611.26
750_2	1769.65	2069.98	14.51	4	1298	3720.85
750_3	1919.34	2128.62	9.83	12	3190	3764.93
750_4	1823.01	1890.67	3.58	13	2565	3925.77
750_5	1609.47	1859.45	13.44	10	2275	3636.81

Again, we analyze the structure of the optimal solutions or best integer feasible solutions found for the two different settings to reach a conclusion on their use. Tables 5.12 and 5.13 present the number of terminals in the solution, the budget and distance limit values, the total revenue and total expenditure for that run. It can be seen the runs may present quite different best integer feasible solutions depending on the setting chosen, with a varying number of terminals in the solution, even if their gaps (presented in Tables 5.10 and 5.11) are similar.

Table 5.12: CG instances | Number of periods: 3 | One budget limit throughout all time horizon | Solution Structure

name	termInSol	budgetLimit	distLimit	totalRev	totalSpent
50_1	3	3665.00	3476.00	1313.56	402.10
50_2	2	4402.00	4233.00	1150.53	150.06

Continued on next page

Table 5.12 – *Continued from previous page*

name	termInSol	budgetLimit	distLimit	totalRev	totalSpent
50_3	1	1804.00	1737.00	586.20	0.00
50_4	1	2758.00	2658.00	598.20	0.00
50_5	5	6914.00	6583.00	2631.49	1011.05
100_1	10	13335.00	12740.00	2517.12	785.40
100_2	9	12537.00	12270.00	1901.18	872.70
100_3	7	9602.00	9172.00	1647.61	612.48
100_4	6	11179.00	10790.00	1512.77	531.99
100_5	6	9251.00	8824.00	1630.34	621.85
250_1	14	23140.00	21995.00	1375.33	685.82
250_2	17	25213.00	24077.00	1601.14	569.78
250_3	15	17271.00	16426.00	1445.28	561.06
250_4	17	30180.00	28803.00	1171.48	555.20
250_5	13	16966.00	16050.00	1186.08	639.98
500_1	27	40638.00	38795.00	1157.71	523.36
500_2	33	40547.00	38597.00	1588.68	673.87
500_3	20	26918.00	25612.00	917.94	420.94
500_4	32	48886.00	46572.00	1359.97	668.76
500_5	33	41209.00	39192.00	1582.16	570.17
750_1	42	64567.00	61483.00	1130.91	526.27
750_2	40	65077.00	62005.00	1192.69	509.10
750_3	50	73446.00	69917.00	1477.58	603.74
750_4	54	72060.00	68730.00	1515.08	564.51
750_5	53	61390.00	58443.00	1558.37	906.94

Table 5.13: CG instances | Number of periods: 3 | A budget limit for each time period | Solution Structure

name	termInSol	budgetLimit	distLimit	totalRev	totalSpent
50_1	3	1222.00	3476.00	1313.56	402.10
50_2	2	1468.00	4233.00	1150.53	150.06
50_3	1	602.00	1737.00	586.20	0.00
50_4	1	920.00	2658.00	598.20	0.00
50_5	5	2305.00	6583.00	2631.49	1011.05
100_1	11	4445.00	12740.00	2698.34	983.95
100_2	10	4179.00	12270.00	2192.76	749.11
100_3	8	3201.00	9172.00	1828.65	797.51

Continued on next page

Table 5.13 – *Continued from previous page*

name	termInSol	budgetLimit	distLimit	totalRev	totalSpent
100_4	6	3727.00	10790.00	1528.89	531.99
100_5	6	3084.00	8824.00	1630.34	621.85
250_1	16	7714.00	21995.00	1526.43	798.55
250_2	18	8405.00	24077.00	1719.11	694.83
250_3	14	5757.00	16426.00	1410.59	535.80
250_4	22	10060.00	28803.00	1591.76	630.42
250_5	13	5656.00	16050.00	1186.08	639.98
500_1	27	13546.00	38795.00	1157.71	523.36
500_2	33	13516.00	38597.00	1451.94	691.86
500_3	20	8973.00	25612.00	917.94	420.94
500_4	31	16296.00	46572.00	1338.99	667.07
500_5	32	13737.00	39192.00	1542.93	806.51
750_1	42	21523.00	61483.00	1130.91	526.27
750_2	40	21693.00	62005.00	1192.69	512.09
750_3	50	24482.00	69917.00	1489.94	603.74
750_4	57	24020.00	68730.00	1748.45	639.03
750_5	53	20464.00	58443.00	1558.37	906.94

Next, we make the comparison from our model's results to Suhl and Hilbert's. Table 5.14 shows their results for a 3-periods run and one budget limit per time period. Comparing to Table 5.11 that presents the same configuration, we see that Suhl and Hilbert's model is unable to find any integer feasible solution for instances with 500 and 750 nodes. For the 250 nodes instances, their model has difficulties finding a good integer feasible solution, but it has success in running the 50 and 100 nodes instances. Further results concerning the comparison of our model and Suhl and Hilbert's are in Appendix A.2. Tables A.29 and A.30 present the results for a 2-periods run, for our model and Suhl and Hilbert's, respectively. Equivalent results are shown for Tables A.31 and A.32, for a 5-periods run.

Table 5.14: CG instances | Number of periods: 3 | One budget limit per period | SUHL AND HILBERT

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	1351.17	1351.17	0.00	210	14	1.36
50_2	1910.28	1910.28	0.00	186	7	1.37

Continued on next page

Table 5.14 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_3	583.62	583.62	0.00	13	0	1.10
50_4	1139.70	1139.70	0.00	1242	10	3.68
50_5	2876.29	2876.41	0.00	25708	27	93.88
100_1	2500.96	2501.14	0.01	216980	47	2903.79
100_2	2318.93	2465.02	5.93	246101	130	3600.06
100_3	2014.72	2014.80	0.00	80446	42	1043.49
100_4	2303.67	2385.98	3.45	225900	90	3600.85
100_5	1861.42	1861.52	0.01	6026	7	68.81
250_1	1667.35	5342.43	68.79	25516	463	3600.18
250_2	1679.01	2044.40	17.87	48101	66	3605.85
250_3	938.29	1254.00	25.18	41501	198	3604.21
250_4	1769.97	2177.25	18.71	37201	200	3601.48
250_5	1089.29	-	-	25510	324	3600.03
500_1	1403.37	-	-	3060	55	3609.71
500_2	1229.33	-	-	2956	66	3600.04
500_3	931.08	-	-	3671	47	3601.29
500_4	1692.59	-	-	2941	68	3639.06
500_5	1318.42	-	-	3111	66	3604.42
750_1	1387.75	-	-	609	13	3600.99
750_2	1406.83	-	-	451	11	3828.95
750_3	1538.77	-	-	871	11	3993.85
750_4	1480.48	-	-	511	16	3628.13
750_5	1254.81	-	-	691	19	3600.70

As mentioned before, results showing different time periods runs are shown in Appendix A.2. For illustrative purposes, Figure 5.2 shows the best integer feasible solution found by our model for a 5-periods run of instance **50_5**. Further information considering this run can be see in Table A.19 in Appendix A.2.

Results comparing the use or not of connectivity constraints are shown in Appendix A.2, in Tables A.18 and A.21 for a 2-periods run and in Tables 5.10 and A.22 for a 3-periods run. Also, full results concerning the use of different separation procedures or both at once are presented in Appendix A.2, discriminated in Tables A.18, A.23 and A.24 for a 2-periods run and in Tables 5.10, A.26 and A.27 for a 3-periods run. Tables A.25 and A.28 summarize these results. Table 5.15 summarizes these results even further, showing simply the percentage of instances better solved by each separation procedure. It is easy to

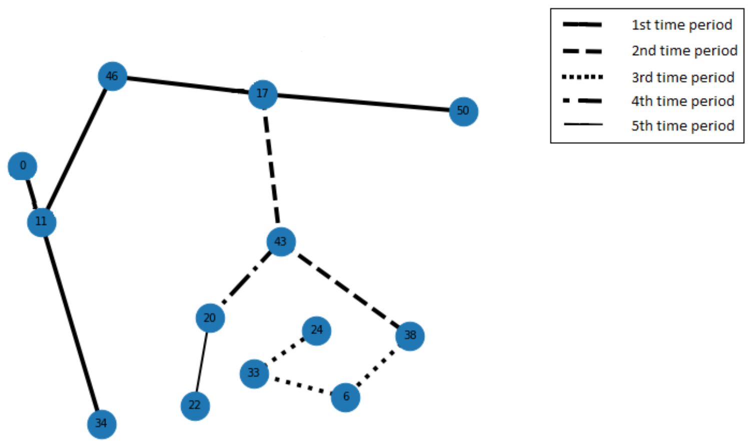


Figure 5.2: Integer feasible solution found for instance **50_5** for a 5-periods run.

see that using either only the integer separation procedure or both separation procedures works best for the CG dataset.

Table 5.15: CG instances | Best results

Sep.	**2 periods(%)**	**3 periods(%)**
Int	56.00	64.00
Both	44.00	32.00
Frac	0.00	4.00

Tables 5.16 presents the results for the same instance through all periods' runs. We can observe the effect of increasing the number of periods in the MPCSTB: the variables increase linearly and so do the constraints, which leads to a directly proportional increase in time. The complexity of the problem does not grow exponentially with the incrementation of the number of periods.

Table 5.16: CG instances | Instance 50_1

# per	**LB**	**UB**	**gap(%)**	**# nodes**	**# cuts**	**time(s)**
2	793.24	793.24	0.00	0	0	0.47
3	1351.17	1351.17	0.00	23	90	1.77
5	2133.01	2133.01	0.00	15	98	2.90
8	3263.26	3263.26	0.00	11	59	6.90

Curiously enough, even if these instances have more variables than the incomplete graph ones, the algorithm takes less time to prove optimality for them. An explanation is that it is harder to find the optimal solution path when not all edges are available to the problem.

5.4 Randomly generated instances of incomplete graphs

These are randomly generated instances emulating real Brazilian gas network expansion instances, that cannot be revealed due to confidentiality agreements. They will be used to test the scalability of our proposed approach and, so, to draw a conclusion concerning the efficiency of the model. The so called "IG instances" have 95% of nodes as Steiner nodes (their profit equals zero) and 5% of nodes have random fractional value profits. We randomly choose edges to be selected in the graph and check it for connectivity, through a connected components algorithm, guaranteeing all nodes are in the same connected component. Furthermore, we tried to generate instances of incomplete graphs with respective densities as low as possible, without losing connectivity. The cost of the selected edges are also fractional and randomly drawn. Since these instances are made of incomplete graphs, they have lesser variables and lesser constraints than complete graph instances. Our first assumption was that incomplete graph instances would be easier to solve due to the lesser amount of variables, however they may be more difficult to solve than complete graph instances, as the path to a profitable node may include a lot of unprofitable ones along the way.

We have generated 5 instances of the same size, to get a clearer look at the model's capabilities. Table 5.17 informs the characteristics of the IG dataset: the name of the instance, the number of vertices, number of edges, number of terminals and percentage of edges in the instance in comparison with the total number of edges of a complete graph, which can be seen in Table 5.9. In Appendix A.3 we can see some general results: Table A.33 presents results for a 3-periods run, Table A.34 shows results for a 5-periods run, Table A.35, for an 8-periods run, Table A.36, for a 10-periods run and, finally, Table A.37, for a 15-periods run. In this Section, we will provide a comparison between different settings for a 2-periods run. Table 5.18 show results for a 2-periods run with only one budget limit throughout all time horizon and Table 5.19 has results for a 2-periods run with one budget limit per time period of the study horizon. Results that follow consider all connectivity constraints and the use of both separation procedures at once. Next, we will compare our results to Suhl and Hilbert's results for a 2-periods run (Table 5.22).

Table 5.17: IG instances

name	$\lvert V\rvert$	$\lvert E\rvert$	**nT**	**perc(%)**
50_1	50	117	3	9.55

Continued on next page

Table 5.17 – *Continued from previous page*

name	$\|V\|$	$\|E\|$	**nT**	
50_2	50	106	3	8.65
50_3	50	137	3	11.18
50_4	50	139	3	11.35
50_5	50	120	3	9.80
100_1	100	245	5	4.95
100_2	100	260	5	5.25
100_3	100	304	5	6.14
100_4	100	261	5	5.27
100_5	100	248	5	5.01
150_1	150	413	7	3.70
150_2	150	409	7	3.66
150_3	150	428	7	3.83
150_4	150	423	7	3.79
150_5	150	435	7	3.89
200_1	200	565	10	2.84
200_2	200	559	10	2.81
200_3	200	552	10	2.77
200_4	200	541	10	2.72
200_5	200	544	10	2.73
250_1	250	837	12	2.69
250_2	250	826	12	2.65
250_3	250	850	12	2.73
250_4	250	828	12	2.66
250_5	250	823	12	2.64
300_1	300	1215	15	2.71
300_2	300	1359	15	3.03
300_3	300	1234	15	2.75
300_4	300	1277	15	2.85
300_5	300	1275	15	2.84

Table 5.18: IG instances| Number of periods: 2 | One budget limit throughout all time horizon

name	**LB**	**UB**	**gap(%)**	**# nodes**	**# cuts**	**time(s)**
50_1	14.86	14.86	0.00	0	0	0.25
50_2	11.91	11.91	0.00	9	60	20.50

Continued on next page

Table 5.18 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_3	14.50	14.50	0.00	7	7	0.39
50_4	14.84	14.84	0.00	17	265	36.11
50_5	4.84	4.84	0.00	14	37	0.56
100_1	16.68	16.68	0.00	25	564	84.69
100_2	20.04	20.04	0.00	71	1059	326.21
100_3	16.95	16.95	0.00	37	1011	47.38
100_4	19.91	19.91	0.00	84	1525	1054.65
100_5	11.83	11.83	0.00	13	185	1.04
150_1	16.60	16.60	0.00	99	1026	356.97
150_2	22.91	22.91	0.00	143	4425	2464.07
150_3	22.15	25.43	12.91	138	4859	3770.14
150_4	22.74	22.74	0.00	179	2653	1676.76
150_5	20.85	25.26	17.46	130	3400	3711.83
200_1	28.23	31.12	9.27	148	5383	3647.14
200_2	26.83	26.83	0.00	106	3792	1095.22
200_3	28.97	28.97	0.00	104	2485	682.51
200_4	24.86	28.28	12.08	188	7640	3642.76
200_5	31.02	31.02	0.00	202	5046	1956.98
250_1	32.27	35.47	9.03	87	7143	3617.63
250_2	30.93	33.99	9.01	224	6872	3768.59
250_3	33.12	35.62	7.03	165	5559	3600.81
250_4	31.63	33.80	6.43	496	5579	3645.78
250_5	34.09	36.20	5.83	267	9732	4045.16
300_1	31.08	43.30	28.21	66	8318	3925.79
300_2	33.89	45.52	25.55	82	7212	3612.93
300_3	39.67	46.46	14.61	71	10878	3601.07
300_4	36.96	42.94	13.93	49	9087	3739.51
300_5	41.26	51.60	20.04	65	10539	3612.27

Table 5.19: IG instances | Number of periods: 2 | A budget limit for each time period

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	14.86	14.86	0.00	0	0	0.26
50_2	11.91	11.91	0.00	11	90	25.51
50_3	14.50	14.50	0.00	5	15	0.38

Continued on next page

Table 5.19 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_4	14.84	14.84	0.00	17	241	0.80
50_5	4.84	4.84	0.00	9	27	5.52
100_1	16.68	16.68	0.00	25	513	137.71
100_2	20.04	20.04	0.00	51	908	173.84
100_3	16.95	16.95	0.00	52	1326	406.82
100_4	19.91	19.91	0.00	114	1949	1826.97
100_5	14.49	14.49	0.00	25	280	6.55
150_1	16.60	16.60	0.00	39	675	235.17
150_2	22.91	22.91	0.00	121	3557	3383.88
150_3	22.55	25.32	10.96	150	4310	3741.13
150_4	22.74	22.74	0.00	159	2086	1305.06
150_5	25.26	25.26	0.00	295	4197	3265.31
200_1	29.90	31.12	3.93	113	4846	3942.10
200_2	26.83	26.83	0.00	69	1852	124.81
200_3	29.50	30.55	3.43	182	5760	3796.22
200_4	28.23	28.23	0.00	141	3724	1602.44
200_5	31.02	31.02	0.00	207	5767	2616.35
250_1	30.79	35.47	13.19	104	9055	3635.50
250_2	30.19	34.98	13.69	140	7048	3872.50
250_3	29.80	38.05	21.69	79	4670	3788.72
250_4	32.54	32.54	0.00	305	4706	864.70
250_5	32.70	36.23	9.75	179	9110	3600.67
300_1	31.07	43.34	28.31	64	6599	3628.63
300_2	35.74	45.52	21.49	56	4526	3605.44
300_3	38.71	44.37	12.76	104	10475	3603.74
300_4	36.62	48.53	24.54	39	6137	3644.26
300_5	39.34	51.60	23.76	61	7059	3651.56

Once again, we are interested in analyzing the structure of the optimal or best feasible integer solution. Tables 5.20 and 5.21 portrait the number of terminals that were able to enter the network, the budget and distance limit values, the total revenue and the total amount spent for that run. A budget limit for the whole horizon generally makes total revenue greater than a budget limit for each time period, because the nodes are allowed to enter the network at an earliest date.

Table 5.20: IG instances | Number of periods: 2 | One budget limit throughout all time horizon | Solution Structure

name	termInSol	budgetLimit	distLimit	totalRev	totalSpent
50_1	1	2.00	1.00	7.88	0.00
50_2	2	2.00	2.00	11.45	0.44
50_3	1	2.00	1.00	7.66	0.00
50_4	1	2.00	2.00	8.00	0.00
50_5	3	2.00	2.00	19.50	0.92
100_1	2	2.00	2.00	10.96	0.32
100_2	2	2.00	2.00	8.35	0.69
100_3	2	2.00	2.00	11.06	0.59
100_4	2	2.00	2.00	8.35	0.68
100_5	3	2.00	2.00	16.44	1.01
150_1	4	2.00	2.00	15.89	0.91
150_2	2	2.00	2.00	9.16	0.21
150_3	2	2.00	2.00	6.87	0.50
150_4	3	2.00	2.00	12.20	1.08
150_5	2	2.00	2.00	6.79	0.55
200_1	3	2.00	2.00	10.07	1.25
200_2	4	2.00	2.00	13.93	0.76
200_3	3	2.00	2.00	11.82	1.07
200_4	3	2.00	2.00	11.84	0.90
200_5	3	2.00	2.00	9.97	1.21
250_1	3	2.00	2.00	11.74	0.79
250_2	4	2.00	2.00	13.68	1.11
250_3	4	2.00	2.00	11.74	0.76
250_4	4	2.00	2.00	13.82	0.44
250_5	4	2.00	2.00	11.69	1.13
300_1	5	3.00	3.00	17.66	2.48
300_2	5	3.00	3.00	15.67	2.79
300_3	5	3.00	3.00	13.86	1.26
300_4	5	3.00	3.00	17.43	2.01
300_5	3	3.00	3.00	9.88	2.82

Table 5.21: IG instances | Number of periods: 2 | A budget limit for each time period | Solution Structure

name	termInSol	budgetLimit	distLimit	totalRev	totalSpent
50_1	1	1.00	1.00	7.88	0.00
50_2	2	1.00	2.00	11.45	0.44
50_3	1	1.00	1.00	7.66	0.00
50_4	1	1.00	2.00	8.00	0.00
50_5	3	1.00	2.00	19.50	0.92
100_1	2	1.00	2.00	10.96	0.32
100_2	2	1.00	2.00	8.35	0.69
100_3	2	1.00	2.00	11.06	0.59
100_4	2	1.00	2.00	8.35	0.68
100_5	3	1.00	2.00	13.77	1.00
150_1	4	1.00	2.00	15.89	0.91
150_2	2	1.00	2.00	9.16	0.21
150_3	2	1.00	2.00	6.87	0.39
150_4	3	1.00	2.00	12.20	1.08
150_5	2	1.00	2.00	6.79	0.55
200_1	3	1.00	2.00	10.07	1.25
200_2	4	1.00	2.00	13.93	0.76
200_3	3	1.00	2.00	9.85	0.68
200_4	3	1.00	2.00	11.84	0.85
200_5	3	1.00	2.00	9.97	1.21
250_1	3	1.00	2.00	11.74	0.79
250_2	3	1.00	2.00	11.76	0.18
250_3	3	1.00	2.00	9.71	1.16
250_4	5	1.00	2.00	15.83	1.19
250_5	4	1.00	2.00	11.66	1.13
300_1	5	2.00	3.00	17.73	2.59
300_2	5	2.00	3.00	15.67	2.79
300_3	5	2.00	3.00	15.82	1.13
300_4	4	2.00	3.00	11.59	1.76
300_5	5	2.00	3.00	16.34	2.66

Subsequently, we compare our model's results with Suhl and Hilbert's. Table 5.22 presents the results they obtain for a 2-periods run and one budget limit per time period. Setting side by side their results and our own (that are in Table 5.19 for this particular configuration), our model and Suhl and

Hilbert's achieve similar performance for the IG dataset. Further results for 1 time period (Table A.47), 3 time periods (Table A.49) and 5 time periods (Table A.51) are shown in Appendix A.3.

Table 5.22: IG instances | Number of periods: 2 | One budget limit per period | SUHL AND HILBERT

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	14.86	14.86	0.00	0	0	0.11
50_2	11.91	11.91	0.00	40	0	0.15
50_3	14.50	14.50	0.00	43	4	0.18
50_4	14.84	14.84	0.00	130	2	0.22
50_5	4.84	4.84	0.00	80	4	0.18
100_1	16.68	16.68	0.00	3228	9	2.77
100_2	20.04	20.04	0.00	7129	38	7.09
100_3	16.95	16.95	0.00	5405	20	6.10
100_4	19.91	19.91	0.00	24082	73	22.58
100_5	14.49	14.49	0.00	461	1	0.73
150_1	16.60	16.60	0.00	35864	51	53.00
150_2	22.91	22.91	0.00	160952	195	299.89
150_3	25.32	25.32	0.00	421264	273	968.93
150_4	22.74	22.74	0.00	153458	105	249.76
150_5	25.26	25.26	0.00	301970	216	528.63
200_1	31.12	31.12	0.00	634807	227	1590.66
200_2	26.83	26.83	0.00	34121	25	54.22
200_3	30.55	30.55	0.00	550574	259	1691.75
200_4	28.23	28.23	0.00	169569	165	326.91
200_5	31.02	31.02	0.00	458155	314	1196.15
250_1	31.49	35.26	10.70	831362	444	3600.04
250_2	29.87	33.73	11.46	772700	684	3600.32
250_3	31.25	35.62	12.28	969900	403	3600.33
250_4	30.36	32.54	6.69	1219477	383	3600.08
250_5	32.51	36.19	10.16	1243891	167	3600.01
300_1	26.56	37.80	29.73	1106257	620	3600.02
300_2	31.68	43.75	27.59	973800	1227	3600.53
300_3	31.30	45.91	31.82	1132949	717	3600.02
300_4	32.85	44.44	26.07	1094226	1148	3600.01
300_5	36.04	47.85	24.67	1004301	986	3600.34

Figure 5.3 gives an illustrative example of the optimal solution found by our model for a 5-periods run of instance (100_4). Table A.34 portraits other information about this run in Appendix A.3.

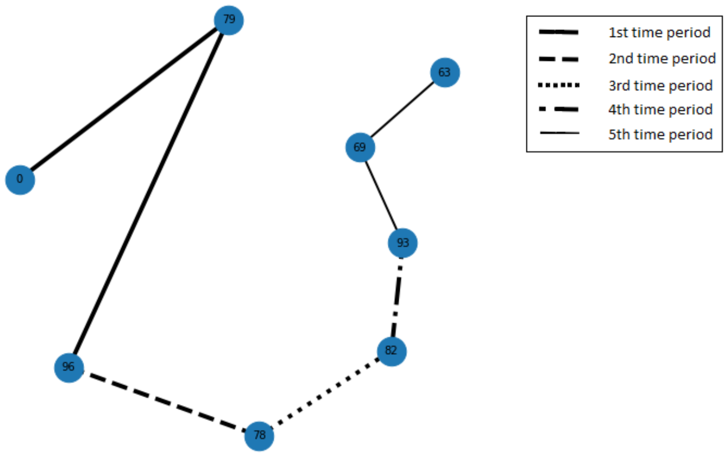


Figure 5.3: Optimal solution found for instance **100_4** for a 5-periods run.

We have established in Section 5.2 that the use of connectivity constraints are underlying to the success of the model. Comparing the results on Tables A.38 and Table A.39 in Appendix A.3, that do not have the valid inequalities in their formulation, to Tables 5.18 and A.33, it is easy to see the improvements those valid inequalities bring to the model.

Another comparison we will attempt to pursue is the difference in results for the use of both separation procedures or just one of them. Again, the complete results are shown in Appendix A.3, but Table 5.23 presents a summary. It is safe to say our regular procedure of using the two separations and all connectivity constraints is the one that works best for the IG dataset.

Table 5.23: IG instances | Best results

Sep.	**2 periods(%)**	**3 periods(%)**
Int	43.33	25.00
Both	53.33	75.00
Frac	3.33	0.00

Table 5.24 presents the results for the same instance through all its periods' runs. From these results, we can clearly see that the complexity of the problem does not grow exponentially with the incrementation of the number of periods.

Table 5.24: IG instances | Instance 100_2

# per	LB	UB	gap(%)	# nodes	# cuts	time(s)
2	20.04	20.04	0.00	71	1059	326.21
3	33.18	33.18	0.00	55	797	244.60
5	51.09	51.09	0.00	100	1623	762.57
8	75.96	75.96	0.00	234	3666	1052.68
10	88.49	88.49	0.00	264	3551	987.62
15	116.23	116.24	0.01	542	7142	2094.73

Chapter 6
Conclusions

The Multi-period Prize-Collecting Steiner Tree problem with Budget constraints (MPCSTB) is a generalization of the classical Prize-Collecting Steiner Tree problem (PCST). The most profitable customers are selected and connected by a least-cost network, along different time periods and respecting a predefined budget and a predefined traveled distance. Therefore, the problem at hand involves planning the expansion of a gas network throughout a multiple number of periods in the near future, considering a distance limit per period and a budget limit per subset set of periods. The objective is to maximize the sum of the profits of the recently incorporated cities reduced by the cost of the new pipeline stretches built.

The aim of this thesis is finding solutions of guaranteed quality for realistic problem sizes in a reasonable amount of computing time. The method of choice is a *branch-and-cut* approach: the cut constraints set is inserted as needed. The use of two separation procedures and a primal heuristic are vital to the success of the model. Benchmark instances from the literature, adapted to a multi-period setting, up to approximately 2000 vertices and 200 terminals, are satisfactorily evaluated with the model. Randomly generated instances up to 750 nodes, represented as complete graphs, and randomly generated incomplete graph instances up to 300 vertices, where approximately 15 are terminals, are satisfactorily evaluated as well. To the best of our knowledge, no other algorithm that attempted to solve the MPCSTB have achieved lower gaps than the ones we have obtained with our algorithm, for the number of periods tested, for instances of these sizes and of this complicated nature.

The MPCSTB considers a multi-period PCST problem with two knapsack constraints (we call it budget and distance constraints). A broader version of the problem would be dealing with several knapsack constraints instead of two. Our problem would become a multi-period prize-collecting Steiner tree problem with multiple knapsack constraints, a harder one, for which we could analyze the impact of increasing this family of inequalities over our ability of providing good solutions. We could also explore the absence of symmetry in the side constraints as this work has the same budget and distance limit values for all subset sets of periods. Future work also includes adding stochasticity to the problem, as the profit parameters considered are not deterministic in practice, but actually uncertain, and predicted through linear regression.

Bibliography

Arulselvan, A., Bley, A., Gollowitzer, S., Ljubić, I., and Maurer, O. (2011). Mip modeling of incremental connected facility location. In *Network Optimization*, pages 490–502. Springer.

Atamtürk, A. (2002). On capacitated network design cut–set polyhedra. *Mathematical Programming*, 92(3):425–437.

Babonneau, F., Nesterov, Y., and Vial, J.-P. (2012). Design and operations of gas transmission networks. *Operations research*, 60(1):34–47.

Bienstock, D., Goemans, M. X., Simchi-Levi, D., and Williamson, D. (1993). A note on the prize collecting traveling salesman problem. *Mathematical programming*, 59(1-3):413–420.

Borraz-Sánchez, C., Bent, R., Backhaus, S., Hijazi, H., and Hentenryck, P. V. (2016). Convex relaxations for gas expansion planning. *INFORMS Journal on Computing*, 28(4):645–656.

Canuto, S. A., Resende, M. G., and Ribeiro, C. C. (2001). Local search with perturbations for the prize-collecting steiner tree problem in graphs. *Networks: An International Journal*, 38(1):50–58.

Chagas, R. J. and Cunha, A. S. (2016). On the np-completeness of the multi-period minimum spanning tree problem. Working paper.

Chagas, R. J., Valle, C. A., and da Cunha, A. S. (2018). Exact solution approaches for the multi-period degree constrained minimum spanning tree problem. *European Journal of Operational Research*.

Cherkassky, B. V. and Goldberg, A. V. (1995). On implementing push-relabel method for the maximum flow problem. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 157–171. Springer.

Chopra, S. and Rao, M. R. (1994a). The steiner tree problem i: Formulations, compositions and extension of facets. *Mathematical Programming*, 64(1-3):209–229.

Chopra, S. and Rao, M. R. (1994b). The steiner tree problem ii: Properties and classes of facets. *Mathematical Programming*, 64(1-3):231–246.

Conforti, M., Cornuéjols, G., and Zambelli, G. (2014). *Integer programming*, volume 271. Springer.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to algorithms*. MIT press.

Costa, A., Cordeau, J.-F., and Laporte, G. (2006). Steiner tree problems with profits. *INFOR: information systems and operational research*, 44(2):99–115.

Costa, A. M., Cordeau, J.-F., and Laporte, G. (2009). Models and branch-and-cut algorithms for the steiner tree problem with revenues, budget and hop constraints. *Networks: An International Journal*, 53(2):141–159.

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

Dantzig, G., Fulkerson, R., and Johnson, S. (1954). Solution of a large-scale traveling-salesman problem. *Journal of the operations research society of America*, 2(4):393–410.

De Wolf, D. and Bakhouya, B. (2012). Optimal dimensioning of pipe networks: the new situation when the distribution and the transportation functions are disconnected. In *Operations Research Proceedings 2011*, pages 369–374. Springer.

De Wolf, D. and Smeers, Y. (1996). Optimal dimensioning of pipe networks with application to gas transmission networks. *Operations Research*, 44(4):596–608.

Elshiekh, T., A Khalil, S., El Mawgoud, H. A., et al. (2013). Optimal design and operation of egyptian gas-transmission pipelines. *Oil and Gas Facilities*, 2(04):44–48.

Feofiloff, P., Fernandes, C. G., Ferreira, C. E., and de Pina, J. C. (2007). Primal-dual approximation algorithms for the prize-collecting steiner tree problem. *Information Processing Letters*, 103(5):195–202.

Feremans, C., Labbé, M., and Laporte, G. (2002). A comparative analysis of several formulations for the generalized minimum spanning tree problem. *Networks: An International Journal*, 39(1):29–34.

Fischetti, M. (1991). Facets of two steiner arborescence polyhedra. *Mathematical Programming*, 51(1-3):401–419.

Fischetti, M., Leitner, M., Ljubić, I., Luipersbeck, M., Monaci, M., Resch, M., Salvagnin, D., and Sinnl, M. (2017). Thinning out steiner trees: a node-based model for uniform edge costs. *Mathematical Programming Computation*, 9(2):203–229.

Ford, L. R. and Fulkerson, D. R. (2009). Maximal flow through a network. In *Classic papers in combinatorics*, pages 243–248. Springer.

Gamrath, G., Koch, T., Maher, S. J., Rehfeldt, D., and Shinano, Y. (2017). Scip-jack—a solver for stp and variants with parallelization extensions. *Mathematical Programming Computation*, 9(2):231–296.

Goemans, M. X. and Myung, Y.-S. (1993). A catalog of steiner tree formulations. *Networks*, 23(1):19–28.

Goemans, M. X. and Williamson, D. P. (1997). The primal-dual method for approximation algorithms and its application to network design problems. *Approximation algorithms for NP-hard problems*, pages 144–191.

Gollowitzer, S. and Ljubić, I. (2011). Mip models for connected facility location: A theoretical and computational study. *Computers & Operations Research*, 38(2):435–449.

Hansen, C. T., Madsen, K., and Nielsen, H. B. (1991). Optimization of pipe networks. *Mathematical Programming*, 52(1-3):45–58.

Humpola, J. and Fügenschuh, A. (2015). Convex reformulations for solving a nonlinear network design problem. *Computational Optimization and Applications*, 62(3):717–759.

Humpola, J., Fügenschuh, A., and Koch, T. (2016). Valid inequalities for the topology optimization problem in gas network design. *OR spectrum*, 38(3):597–631.

IBM (2017). Ilog cplex 12.7.1. *CPLEX User's Manual*.

Johnson, D. S., Minkoff, M., and Phillips, S. (2000). The prize collecting steiner tree problem: theory and practice. In *SODA*, volume 10, page 4. Citeseer.

Kabirian, A. and Hemmati, M. R. (2007). A strategic planning model for natural gas transmission networks. *Energy policy*, 35(11):5656–5670.

Karp, R. M. (1972). Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer.

Kawatra, R. and Bricker, D. (2000). A multiperiod planning model for the capacitated minimal spanning tree problem. *European Journal of Operational Research*, 121(2):412–419.

Klau, G. W., Ljubić, I., Moser, A., Mutzel, P., Neuner, P., Pferschy, U., Raidl, G., and Weiskircher, R. (2004). Combining a memetic algorithm with integer programming to solve the prize-collecting steiner tree problem. In *Genetic and Evolutionary Computation Conference*, pages 1304–1315. Springer.

Ljubić, I., Weiskircher, R., Pferschy, U., Klau, G. W., Mutzel, P., and Fischetti, M. (2005). Solving the prize-collecting steiner tree problem to optimality. In *ALENEX/ANALCO*, pages 68–76.

Ljubić, I., Weiskircher, R., Pferschy, U., Klau, G. W., Mutzel, P., and Fischetti, M. (2006). An algorithmic framework for the exact solution of the prize-collecting steiner tree problem. *Mathematical programming*, 105(2-3):427–449.

Lucena, A. and Resende, M. G. (2004). Strong lower bounds for the prize collecting steiner problem in graphs. *Discrete Applied Mathematics*, 141(1-3):277–294.

Magnanti, T. L. and Raghavan, S. (2005). Strong formulations for network design problems with connectivity requirements. *Networks: An International Journal*, 45(2):61–79.

Magnanti, T. L. and Wolsey, L. A. (1995). Optimal trees. *Handbooks in operations research and management science*, 7:503–615.

Pahl, J., Reiners, T., and Voß, S. (2011). *Network Optimization: 5th International Conference, INOC 2011, Hamburg, Germany, June 13-16, 2011, Proceedings*, volume 6701. Springer.

Poss, M. (2012). Models and algorithms for network design problems. *4OR*, 10(2):215–216.

Rosseti, I., De Aragão, M. P., Ribeiro, C. C., Uchoa, E., and Werneck, R. F. (2003). New benchmark instances for the steiner problem in graphs. In *Metaheuristics: Computer Decision-Making*, pages 601–614. Springer.

Soliman, F. and Murtagh, B. (1982). The solution of large-scale gas pipeline design problems. *Engineering Optimization*, 6(2):77–83.

Suhl, U. H. and Hilbert, H. (1998). A branch-and-cut algorithm for solving generalized multiperiod steiner problems in graphs. *Networks: An International Journal*, 31(4):273–282.

Uchoa, E. (2006). Reduction tests for the prize-collecting steiner problem. *Operations Research Letters*, 34(4):437–444.

Üster, H. and Dilaveroğlu, Ş. (2014). Optimization for design and operation of natural gas transmission networks. *Applied Energy*, 133:56–69.

Appendix A
Complete results

A.1
Modified PUCNU instances

Tables A.1. A.2 and A.3 show results of 2-periods, 3-periods and 5-periods runs, respectively. These results have been run using all connectivity constraints and both separation procedures. There is one budget limit throughout all time horizon.

Table A.1: Modified PUCNU instances | Number of periods: 2 | One budget limit throughout all time horizon

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	478.00	478.00	0.00	10	665	787.76
bip52nu	464.91	470.00	1.08	28	1195	3729.48
bip62nu	450.98	454.00	0.66	36	2298	3655.48
bipa2nu	659.42	673.00	2.02	4	668	3781.63
bipe2nu	109.00	109.00	0.00	447	2560	2833.29
cc10-2nu	322.67	345.00	6.47	12	4014	3605.14
cc11-2nu	597.22	625.00	4.44	6	3076	3617.59
cc12-2nu	1112.62	1180.00	5.71	0	0	3714.39
cc3-10nu	105.84	117.00	9.54	49	5957	3605.60
cc3-11nu	151.49	169.00	10.36	17	3461	3608.81
cc3-12nu	182.00	197.00	7.61	28	3578	3614.77
cc3-4nu	19.00	19.00	0.00	0	0	0.35
cc3-5nu	32.00	32.00	0.00	46	586	291.62
cc5-3nu	67.00	67.00	0.00	49	2916	1218.64
cc6-2nu	27.00	27.00	0.00	27	726	1348.18
cc6-3nu	182.61	196.00	6.83	40	5806	3761.66
cc7-3nu	540.12	574.00	5.90	4	1651	3620.50
cc9-2nu	153.85	158.00	2.63	36	6674	3601.91

Table A.2: Modified PUCNU instances | Number of periods: 3 | One budget limit throughout all time horizon

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	693.41	702.00	1.22	12	1478	4059.78
bip52nu	675.11	678.00	0.43	83	3218	3708.23
bip62nu	654.69	658.00	0.50	29	1564	4035.06
bipa2nu	963.02	995.00	3.21	0	0	3656.65
bipe2nu	156.00	156.00	0.00	17	470	101.11
cc10-2nu	467.55	503.00	7.05	26	6928	3606.31
cc11-2nu	869.60	906.00	4.02	4	2108	3607.32
cc12-2nu	1623.73	1694.00	4.15	0	0	3694.03
cc3-10nu	152.08	169.00	10.01	17	3194	3606.79
cc3-11nu	218.10	241.00	9.50	13	2823	3610.98
cc3-12nu	263.50	288.00	8.51	16	2397	3617.92
cc3-4nu	26.00	26.00	0.00	25	593	565.32
cc3-5nu	43.00	43.00	0.00	3	135	2.43
cc5-3nu	92.14	100.00	7.86	31	2209	3885.70
cc6-2nu	39.00	39.00	0.00	30	972	1761.96
cc6-3nu	264.53	287.00	7.83	14	3293	3603.92
cc7-3nu	789.71	842.00	6.21	4	1494	3626.84
cc9-2nu	221.99	238.00	6.73	20	3232	3602.83

Table A.3: Modified PUCNU instances | Number of periods: 5 | One budget limit throughout all time horizon

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	1155.08	1161.00	0.51	11	1455	3621.12
bip52nu	1125.37	1130.00	0.41	20	1425	3634.48
bip62nu	1095.38	1104.00	0.78	44	3027	3728.42
bipa2nu	1596.22	1641.00	2.73	0	0	3693.46
bipe2nu	263.51	267.00	1.31	251	1772	3607.58
cc10-2nu	767.69	813.00	5.57	14	4401	3628.37
cc11-2nu	1427.20	1489.00	4.15	3	1807	3631.46
cc12-2nu	-	-	-	-	-	Memout
cc3-10nu	254.24	284.00	10.48	14	2694	3609.50
cc3-11nu	352.41	392.00	10.10	9	1880	3616.17
cc3-12nu	432.00	470.00	8.09	7	1460	3625.94

Continued on next page

Table A.3 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
cc3-4nu	44.00	44.00	0.00	23	484	999.08
cc3-5nu	65.23	74.00	11.85	47	2139	3600.15
cc5-3nu	150.15	163.00	7.89	46	2555	3602.16
cc6-2nu	52.94	58.00	8.73	37	1546	3609.47
cc6-3nu	431.82	463.00	6.73	9	2441	4220.98
cc7-3nu	1288.01	1365.00	5.64	2	1036	3635.87
cc9-2nu	359.05	374.00	4.00	14	3897	3603.24

Tables A.4 and A.5 show results of runs without valid inequalities that serve as important connectivity constraints. We can see in Tables A.4 and A.5 that the gaps are substantially higher if compared to gaps in Tables A.1 and A.2.

Table A.4: Modified PUCNU instances | Number of periods: 2 | One budget limit throughout all time horizon | Without connectivity constraints

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	446.94	486.00	8.04	93	15330	3650.74
bip52nu	430.00	518.00	16.99	38	19405	4013.59
bip62nu	432.00	510.00	15.29	34	21699	3685.02
bipa2nu	621.00	759.00	18.18	14	31010	3618.76
bipe2nu	103.00	111.00	7.21	147	19305	3624.98
cc10-2nu	280.00	368.00	23.91	170	22550	3603.41
cc11-2nu	526.00	688.00	23.55	129	26198	3610.76
cc12-2nu	989.00	1261.00	21.57	77	30821	3642.12
cc3-10nu	100.50	113.00	11.06	235	18057	3603.59
cc3-11nu	146.00	175.00	16.57	204	21207	3605.21
cc3-12nu	174.14	213.00	18.25	254	24151	3600.65
cc3-4nu	19.00	19.00	0.00	19	402	2.37
cc3-5nu	32.00	32.00	0.00	114	2497	48.83
cc5-3nu	60.00	69.00	13.04	177	8770	3912.89
cc6-2nu	27.00	27.00	0.00	61	902	7.89
cc6-3nu	169.45	209.00	18.92	227	20866	3601.99
cc7-3nu	483.00	627.00	22.97	164	28192	3613.36
cc9-2nu	131.67	175.00	24.76	138	10753	3647.92

Table A.5: Modified PUCNU instances | Number of periods: 3 | One budget limit throughout all time horizon | Without connectivity constraints

name	**LB**	**UB**	**gap(%)**	**# nodes**	**# cuts**	**time(s)**
bip42nu	642.00	720.00	10.83	76	19872	3959.02
bip52nu	632.79	760.00	16.74	40	24274	3613.04
bip62nu	624.00	748.00	16.58	33	33896	3718.69
bipa2nu	903.00	1103.00	18.13	41	39572	3601.68
bipe2nu	154.01	156.00	1.27	118	19219	3639.23
cc10-2nu	407.00	535.00	23.93	142	22855	3603.93
cc11-2nu	768.00	977.00	21.39	110	26681	3611.90
cc12-2nu	1451.00	1869.00	22.36	55	30041	3644.99
cc3-10nu	137.00	181.00	24.31	200	18163	3603.70
cc3-11nu	206.50	251.00	17.73	215	21368	3605.86
cc3-12nu	230.00	308.00	25.32	201	21042	3609.53
cc3-4nu	26.00	26.00	0.00	127	2549	1279.84
cc3-5nu	43.00	43.00	0.00	173	4686	605.00
cc5-3nu	82.00	103.00	20.39	157	9629	3792.40
cc6-2nu	39.00	39.00	0.00	90	2357	306.05
cc6-3nu	232.13	300.00	22.62	145	17234	3623.73
cc7-3nu	705.00	902.00	21.84	92	26345	3613.53
cc9-2nu	197.25	249.00	20.78	160	16137	3601.33

Table A.6 shows the results for a 2-periods run, when the model uses only the separation of integer infeasible solutions. Table A.9 is the same, but for a 3-periods run. Table A.7 shows the results where the model uses only the separation of fractional infeasible solutions for a 2-periods run and Table A.10 is the same, but for a 3-periods run. Tables A.8 and A.11 show a summary of these results.

Table A.6: Modified PUCNU instances | Number of periods: 2 | One budget limit throughout all time horizon | Separation of integer infeasible solutions

name	**LB**	**UB**	**gap(%)**	**# nodes**	**# cuts**	**time(s)**
bip42nu	478.00	478.00	0.00	10	74	226.30
bip52nu	466.00	466.00	0.00	1590	92	2451.69
bip62nu	451.27	454.00	0.60	1349	48	3600.52
bipa2nu	659.57	679.00	2.86	0	0	3646.97
bipe2nu	109.00	109.00	0.00	496	0	86.75

Continued on next page

Table A.6 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
cc10-2nu	322.84	338.00	4.48	854	854	3605.20
cc11-2nu	596.67	623.00	4.23	322	1079	3601.17
cc12-2nu	1112.62	1180.00	5.71	0	0	3720.97
cc3-10nu	106.44	115.00	7.44	8174	1133	3601.48
cc3-11nu	150.64	167.00	9.80	3271	500	3601.51
cc3-12nu	182.00	192.00	5.21	1306	452	3603.44
cc3-4nu	19.00	19.00	0.00	0	0	0.35
cc3-5nu	32.00	32.00	0.00	84	28	2.83
cc5-3nu	67.00	67.00	0.00	285	13	13.52
cc6-2nu	27.00	27.00	0.00	36	4	1.19
cc6-3nu	182.50	193.00	5.44	2899	842	3603.48
cc7-3nu	539.10	568.00	5.09	307	690	3602.65
cc9-2nu	155.33	157.00	1.07	5790	808	3602.08

Table A.7: Modified PUCNU instances | Number of periods: 2 | One budget limit throughout all time horizon | Separation of fractional infeasible solutions

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	478.00	478.00	0.00	10	994	1724.36
bip52nu	464.91	470.00	1.08	31	1252	3665.95
bip62nu	451.08	454.00	0.64	44	2312	3698.76
bipa2nu	659.60	673.00	1.99	3	495	3796.17
bipe2nu	109.00	109.00	0.00	447	2560	2834.63
cc10-2nu	322.62	345.00	6.49	11	3817	3604.86
cc11-2nu	597.22	625.00	4.44	6	3076	3618.23
cc12-2nu	1112.62	1180.00	5.71	0	0	3722.35
cc3-10nu	105.59	117.00	9.76	18	2881	3605.87
cc3-11nu	150.97	169.00	10.67	15	2776	3608.57
cc3-12nu	182.00	197.00	7.61	34	3831	3614.37
cc3-4nu	19.00	19.00	0.00	0	0	0.29
cc3-5nu	32.00	32.00	0.00	49	672	462.29
cc5-3nu	67.00	67.00	0.00	49	2916	1218.13
cc6-2nu	27.00	27.00	0.00	19	659	547.89
cc6-3nu	182.75	196.00	6.76	32	6025	3643.93
cc7-3nu	539.90	574.00	5.94	4	1443	3620.23
cc9-2nu	153.75	158.00	2.69	37	5884	3601.86

Table A.8: Modified PUCNU instances | Number of periods: 2 | Best results

name	LB	UB	gap(%)	# nodes	# cuts	time(s)	sep.
bip42nu	478.00	478.00	0.00	10	665	787.76	Both
bip52nu	466.00	466.00	0.00	1590	92	2451.69	Int
bip62nu	451.27	454.00	0.60	1349	48	3600.52	Int
bipa2nu	659.60	673.00	1.99	3	495	3796.17	Frac
bipe2nu	109.00	109.00	0.00	447	2560	2833.29	Both
cc10-2nu	322.84	338.00	4.48	854	854	3605.20	Int
cc11-2nu	596.67	623.00	4.23	322	1079	3601.17	Int
cc12-2nu	1112.62	1180.00	5.71	0	0	3714.39	Both
cc3-10nu	106.44	115.00	7.44	8174	1133	3601.48	Int
cc3-11nu	150.64	167.00	9.80	3271	500	3601.51	Int
cc3-12nu	182.00	192.00	5.21	1306	452	3603.44	Int
cc3-4nu	19.00	19.00	0.00	0	0	0.35	Both
cc3-5nu	32.00	32.00	0.00	46	586	291.62	Both
cc5-3nu	67.00	67.00	0.00	49	2916	1218.64	Both
cc6-2nu	27.00	27.00	0.00	27	726	1348.18	Both
cc6-3nu	182.50	193.00	5.44	2899	842	3603.48	Int
cc7-3nu	539.10	568.00	5.09	307	690	3602.65	Int
cc9-2nu	155.33	157.00	1.07	5790	808	3602.08	Int

Table A.9: Modified PUCNU instances | Number of periods: 3 | One budget limit throughout all time horizon | Separation of integer infeasible solutions

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	695.00	695.00	0.00	232	91	475.03
bip52nu	675.16	678.00	0.42	184	91	3602.04
bip62nu	654.74	658.00	0.50	573	92	3603.65
bipa2nu	963.02	995.00	3.21	0	0	3656.37
bipe2nu	156.00	156.00	0.00	15	20	73.38
cc10-2nu	467.90	493.00	5.09	359	622	3606.10
cc11-2nu	868.62	903.00	3.81	231	615	3604.94
cc12-2nu	1623.73	1694.00	4.15	0	0	3692.63
cc3-10nu	152.44	164.00	7.05	3415	786	3600.63
cc3-11nu	217.56	237.00	8.20	1041	420	3601.39
cc3-12nu	263.50	282.00	6.56	623	455	3602.29

Continued on next page

Table A.9 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
cc3-4nu	26.00	26.00	0.00	47	4	2.98
cc3-5nu	43.00	43.00	0.00	21	27	3.45
cc5-3nu	96.00	96.00	0.00	3440	84	789.07
cc6-2nu	39.00	39.00	0.00	92	22	4.11
cc6-3nu	263.98	278.00	5.04	2368	1298	3603.62
cc7-3nu	788.51	827.00	4.65	258	813	3609.24
cc9-2nu	223.04	231.00	3.44	3422	463	3600.10

Table A.10: Modified PUCNU instances | Number of periods: 3 | One budget limit throughout all time horizon | Separation of fractional infeasible solutions

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	693.46	698.00	0.65	84	1504	3653.97
bip52nu	675.11	678.00	0.43	63	2589	4015.07
bip62nu	654.65	658.00	0.51	35	1250	3795.67
bipa2nu	963.02	995.00	3.21	0	0	3656.44
bipe2nu	156.00	156.00	0.00	19	454	108.18
cc10-2nu	467.56	503.00	7.05	14	3693	3604.38
cc11-2nu	869.84	906.00	3.99	4	2618	3621.96
cc12-2nu	1623.73	1694.00	4.15	0	0	3694.71
cc3-10nu	152.08	169.00	10.01	17	3194	3606.81
cc3-11nu	218.17	241.00	9.47	13	2379	3610.94
cc3-12nu	263.50	288.00	8.51	13	1828	3617.79
cc3-4nu	26.00	26.00	0.00	15	466	409.18
cc3-5nu	43.00	43.00	0.00	5	160	3.00
cc5-3nu	92.59	99.00	6.48	62	7314	4334.14
cc6-2nu	39.00	39.00	0.00	38	884	1207.81
cc6-3nu	264.53	287.00	7.83	14	3293	3604.36
cc7-3nu	789.71	842.00	6.21	4	1494	3627.45
cc9-2nu	222.87	238.00	6.36	19	4806	3602.19

Table A.11: Modified PUCNU instances | Number of periods: 3 | Best results

name	LB	UB	gap(%)	# nodes	# cuts	time(s)	sep.
bip42nu	695.00	695.00	0.00	232	91	475.03	Int

Continued on next page

Table A.11 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)	sep.
bip52nu	675.16	678.00	0.42	184	91	3602.04	Int
bip62nu	654.69	658.00	0.50	29	1564	4035.06	Both
bipa2nu	963.02	995.00	3.21	0	0	3656.65	Both
bipe2nu	156.00	156.00	0.00	17	470	101.11	Both
cc10-2nu	467.90	493.00	5.09	359	622	3606.10	Int
cc11-2nu	868.62	903.00	3.81	231	615	3604.94	Int
cc12-2nu	1623.73	1694.00	4.15	0	0	3694.03	Both
cc3-10nu	152.44	164.00	7.05	3415	786	3600.63	Int
cc3-11nu	217.56	237.00	8.20	1041	420	3601.39	Int
cc3-12nu	263.50	282.00	6.56	623	455	3602.29	Int
cc3-4nu	26.00	26.00	0.00	25	593	565.32	Both
cc3-5nu	43.00	43.00	0.00	3	135	2.43	Both
cc5-3nu	96.00	96.00	0.00	3440	84	789.07	Int
cc6-2nu	39.00	39.00	0.00	30	972	1761.96	Both
cc6-3nu	263.98	278.00	5.04	2368	1298	3603.62	Int
cc7-3nu	788.51	827.00	4.65	258	813	3609.24	Int
cc9-2nu	223.04	231.00	3.44	3422	463	3600.10	Int

Tables A.12 and A.13 show the results for our model and Suhl and Hilbert's, respectively, for a 1-period run. Tables A.14 and A.15 show the results for a 2-periods run and, finally, Tables A.16 and A.17 show the results for a 3-periods run.

Table A.12: Modified PUCNU instances | Number of periods: 1 | One budget limit per period

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	258.00	258.00	0.00	10	787	893.32
bip52nu	252.00	252.00	0.00	12	826	1819.42
bip62nu	242.62	244.00	0.57	57	2838	3722.17
bipa2nu	356.46	367.00	2.87	4	740	3602.01
bipe2nu	59.00	59.00	0.00	0	0	4.88
cc10-2nu	175.64	188.00	6.57	25	7126	3604.29
cc11-2nu	324.08	339.00	4.40	8	3693	3613.74
cc12-2nu	605.33	630.00	3.92	2	1731	3654.62
cc3-10nu	58.54	66.00	11.30	77	6503	3605.05
cc3-11nu	82.37	91.00	9.48	25	3843	3603.87

Continued on next page

Table A.12 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
cc3-12nu	99.00	108.00	8.33	36	3253	3702.26
cc3-4nu	10.00	10.00	0.00	0	0	0.23
cc3-5nu	18.00	18.00	0.00	1	25	1.20
cc5-3nu	37.00	37.00	0.00	79	7990	2846.63
cc6-2nu	15.00	15.00	0.00	0	0	0.21
cc6-3nu	99.68	104.00	4.16	36	7163	3602.84
cc7-3nu	293.59	309.00	4.99	8	3162	3615.62
cc9-2nu	86.00	86.00	0.00	25	1813	962.91

Table A.13: Modified PUCNU instances | Number of periods: 1 | One budget limit per period | SUHL AND HILBERT

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	258.00	258.00	0.00	12706	28	555.34
bip52nu	252.00	252.00	0.00	42283	23	2715.79
bip62nu	242.67	244.00	0.55	59746	106	3600.12
bipa2nu	355.10	436.00	18.56	190	26	3693.08
bipe2nu	59.00	59.00	0.00	46	9	25.00
cc10-2nu	174.25	181.00	3.73	461688	509	3600.08
cc11-2nu	320.71	348.00	7.84	18461	345	3602.82
cc12-2nu	598.88	699.00	14.32	682	58	3825.44
cc3-10nu	59.00	64.00	7.81	19473	567	3601.10
cc3-11nu	82.00	93.00	11.83	1800	279	3609.34
cc3-12nu	99.00	111.00	10.81	1276	139	3606.92
cc3-4nu	10.00	10.00	0.00	0	0	0.15
cc3-5nu	18.00	18.00	0.00	5	0	0.36
cc5-3nu	37.00	37.00	0.00	1451	22	4.55
cc6-2nu	15.00	15.00	0.00	23	1	0.22
cc6-3nu	99.25	102.00	2.70	534196	239	3600.04
cc7-3nu	290.50	337.00	13.80	905	154	3604.76
cc9-2nu	86.00	86.00	0.00	265882	140	959.87

Table A.14: Modified PUCNU instances | Number of periods: 2 | One budget limit per period

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	502.50	504.00	0.30	30	2105	3824.44
bip52nu	490.32	498.00	1.54	7	1020	4100.66
bip62nu	478.31	480.00	0.35	70	1891	3748.53
bipa2nu	697.24	712.00	2.07	2	563	3912.01
bipe2nu	112.00	112.00	0.00	2	93	34.17
cc10-2nu	336.85	356.00	5.38	21	4841	3620.07
cc11-2nu	623.65	648.00	3.76	4	2485	3618.13
cc12-2nu	1164.91	1213.00	3.96	0	0	3713.87
cc3-10nu	109.54	120.00	8.72	56	6456	3605.64
cc3-11nu	156.73	172.00	8.88	14	2737	3608.94
cc3-12nu	189.00	202.00	6.44	26	3705	3614.81
cc3-4nu	19.00	19.00	0.00	8	259	2.56
cc3-5nu	32.00	32.00	0.00	0	54	1.45
cc5-3nu	69.00	69.00	0.00	25	1667	362.96
cc6-2nu	27.00	27.00	0.00	20	585	1023.52
cc6-3nu	189.74	194.00	2.20	22	4463	3603.73
cc7-3nu	563.88	590.00	4.43	4	1435	3621.04
cc9-2nu	158.82	170.00	6.58	31	7772	3602.02

Table A.15: Modified PUCNU instances | Number of periods: 2 | One budget limit per period | SUHL AND HILBERT

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	498.79	504.00	1.03	43600	3	3611.74
bip52nu	487.34	493.00	1.15	9101	14	3643.40
bip62nu	476.96	482.00	1.05	7801	50	3604.11
bipa2nu	694.29	706.00	1.66	759	5	3600.71
bipe2nu	112.00	112.00	0.00	38	0	28.47
cc10-2nu	330.21	349.00	5.38	144211	477	3600.09
cc11-2nu	611.13	720.00	15.12	6101	218	3628.87
cc12-2nu	1142.88	1398.00	18.25	701	36	3600.21
cc3-10nu	108.38	122.00	11.17	4817	271	3605.49
cc3-11nu	156.00	186.00	16.13	2500	175	3602.55
cc3-12nu	186.90	212.00	11.84	2200	83	3605.93

Continued on next page

Table A.15 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
cc3-4nu	19.00	19.00	0.00	5	0	0.24
cc3-5nu	32.00	32.00	0.00	164	2	1.19
cc5-3nu	69.00	69.00	0.00	7773	36	27.93
cc6-2nu	27.00	27.00	0.00	105	0	0.36
cc6-3nu	187.63	198.00	5.24	255601	378	3600.33
cc7-3nu	553.89	-	-	2278	140	3626.81
cc9-2nu	158.24	162.00	2.32	501291	194	3600.05

Table A.16: Modified PUCNU instances | Number of periods: 3 | One budget limit per period

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	742.89	751.00	1.08	26	1437	3676.89
bip52nu	725.66	728.00	0.32	29	1888	3754.22
bip62nu	709.08	716.00	0.97	20	2232	3893.25
bipa2nu	1041.47	1068.00	2.48	0	0	3657.31
bipe2nu	164.00	167.00	1.80	90	2404	3608.39
cc10-2nu	493.19	518.00	4.79	12	4238	3688.91
cc11-2nu	922.09	957.00	3.65	4	2030	3622.01
cc12-2nu	1727.01	1804.00	4.27	0	0	3806.25
cc3-10nu	159.62	175.00	8.79	29	2726	3602.13
cc3-11nu	231.48	251.00	7.78	15	3140	3610.92
cc3-12nu	276.00	300.00	8.00	8	1184	3617.33
cc3-4nu	26.00	26.00	0.00	27	707	537.10
cc3-5nu	43.00	43.00	0.00	13	477	10.90
cc5-3nu	95.75	101.00	5.19	33	4619	3619.86
cc6-2nu	39.00	39.00	0.00	48	989	588.50
cc6-3nu	276.37	293.00	5.68	10	2729	3604.04
cc7-3nu	839.06	877.00	4.33	3	1827	3625.20
cc9-2nu	234.27	240.00	2.39	22	5837	3602.67

Table A.17: Modified PUCNU instances | Number of periods: 3 | One budget limit per period | SUHL AND HILBERT

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
bip42nu	732.15	746.00	1.86	22040	8	3600.63
bip52nu	716.44	738.00	2.92	4300	7	3756.00
bip62nu	703.03	718.00	2.08	2640	12	3601.01
bipa2nu	1031.09	1308.00	21.17	84	7	3602.91
bipe2nu	162.58	167.00	2.64	18915	0	3600.06
cc10-2nu	479.43	513.00	6.54	86052	326	3600.12
cc11-2nu	893.35	-	-	18161	169	3619.55
cc12-2nu	1676.49	-	-	4105	38	3663.06
cc3-10nu	156.64	182.00	13.93	6240	244	3605.96
cc3-11nu	225.65	279.00	19.12	4400	116	3602.00
cc3-12nu	269.82	324.00	16.72	2200	46	3811.31
cc3-4nu	26.00	26.00	0.00	402	0	0.95
cc3-5nu	43.00	43.00	0.00	1432	27	8.00
cc5-3nu	100.00	100.00	0.00	83401	38	406.41
cc6-2nu	39.00	39.00	0.00	239	0	0.65
cc6-3nu	270.73	289.00	6.32	112500	311	3600.18
cc7-3nu	817.27	-	-	8415	150	3626.53
cc9-2nu	229.54	241.00	4.76	346801	81	3600.28

A.2
Randomly generated instances of complete graphs

Firstly, Tables A.18, A.19, A.20 show the results for all five instances of each instance size set, considering all connectivity constraints and all separation procedures, for a 2-periods, 5-periods and 8-periods runs, respectively.

Table A.18: CG instances| Number of periods: 2 | One budget limit throughout all time horizon

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	793.24	793.24	0.00	0	0	0.47
50_2	1323.54	1323.54	0.00	11	88	1.01
50_3	389.08	389.08	0.00	0	0	0.51
50_4	656.58	656.58	0.00	3	26	0.79

Continued on next page

Table A.18 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_5	2055.44	2055.44	0.00	67	622	643.06
100_1	1804.10	1804.10	0.00	103	1011	667.86
100_2	1737.99	1737.99	0.00	274	1989	2593.10
100_3	1431.96	1559.36	8.17	85	1868	3682.62
100_4	1613.58	1613.58	0.00	49	381	189.48
100_5	1351.02	1351.02	0.00	27	132	28.37
250_1	1444.15	1526.15	5.37	36	2467	3620.57
250_2	1424.74	1445.49	1.44	69	2901	3665.88
250_3	896.93	907.65	1.18	115	3106	3611.94
250_4	1517.34	1535.60	1.19	110	2773	3604.12
250_5	988.65	1024.94	3.54	67	1881	3892.93
500_1	1233.17	1257.63	1.94	36	2343	3693.54
500_2	1144.73	1195.02	4.21	25	2434	3749.33
500_3	855.97	1091.66	21.59	20	2588	4167.29
500_4	1486.20	1509.00	1.51	12	1676	3658.88
500_5	1147.48	1158.42	0.94	64	2947	3643.31
750_1	1262.61	1363.15	7.38	12	1591	3634.73
750_2	1268.21	1285.43	1.34	12	2004	3605.74
750_3	1409.48	1453.31	3.02	19	2939	3622.45
750_4	1297.52	1303.43	0.45	39	2757	3865.12
750_5	1222.07	1261.19	3.10	17	2351	3716.72

Table A.19: CG instances| Number of periods: 5 | One budget limit throughout all time horizon

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	6264.45	6264.45	0.00	7	111	20.71
50_2	11567.85	11567.85	0.00	33	310	111.91
50_3	2918.10	2918.10	0.00	0	34	12.78
50_4	5698.50	5698.50	0.00	0	70	217.69
50_5	12563.53	17277.21	27.28	174	3547	3607.45

Table A.20: CG instances| Number of periods: 8 | One budget limit throughout all time horizon

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	3263.26	3263.26	0.00	11	59	6.90
50_2	6169.52	6169.52	0.00	31	445	98.99
50_3	1556.32	1556.32	0.00	0	43	3.79
50_4	3039.20	3039.20	0.00	25	199	16.53
50_5	5781.91	7946.37	27.24	126	2668	3604.94

Tables A.21 and A.22 show results of runs without valid inequalities that serve as important connectivity constraints. Hence, gaps are higher if compared to Tables A.18 and 5.10.

Table A.21: CG instances | Number of periods: 2 | One budget limit throughout all time horizon | Without connectivity constraints

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	793.24	793.24	0.00	6	143	1.26
50_2	1323.54	1323.54	0.00	35	268	6.17
50_3	389.08	389.08	0.00	5	122	0.86
50_4	656.58	656.58	0.00	27	227	1.29
50_5	2055.44	2055.44	0.00	430	2851	2679.45
100_1	1568.20	2235.62	29.85	141	4706	3627.57
100_2	1642.74	1761.51	6.74	216	4461	3643.96
100_3	1375.10	1556.54	11.66	256	4530	3627.03
100_4	1464.30	1694.88	13.60	191	4952	3723.59
100_5	1351.02	1351.02	0.00	310	4550	2259.38
250_1	1142.15	1549.37	26.28	144	8143	3655.26
250_2	1113.49	1664.84	33.12	194	9309	3687.43
250_3	836.95	968.96	13.62	252	7710	3710.98
250_4	1442.45	1632.34	11.63	242	12996	3629.74
250_5	725.91	1369.04	46.98	98	5835	3767.41
500_1	968.61	1428.38	32.19	209	11878	3647.19
500_2	978.41	1307.00	25.14	169	13129	3669.31
500_3	773.95	915.33	15.45	202	15878	3604.01
500_4	1290.63	1744.95	26.04	138	13336	3604.14
500_5	928.55	1711.34	45.74	196	10554	3784.27
750_1	1038.82	1549.05	32.94	117	17184	3742.34

Continued on next page

Table A.21 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
750_2	1003.54	1807.06	44.47	126	18325	3606.35
750_3	1131.99	1983.22	42.92	184	27349	3624.87
750_4	1042.64	1679.09	37.90	92	10886	3808.70
750_5	1015.52	1647.28	38.35	114	16276	3828.12

Table A.22: CG instances | Number of periods: 3 | One budget limit throughout all time horizon | Without connectivity constraints

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	1351.17	1351.17	0.00	71	333	7.00
50_2	1910.28	1910.28	0.00	60	674	17.12
50_3	583.62	583.62	0.00	3	199	1.45
50_4	1139.70	1139.70	0.00	81	515	17.05
50_5	2128.69	3411.22	37.60	165	2732	3605.80
100_1	1919.51	2646.07	27.46	119	4045	3676.87
100_2	1556.73	3152.78	50.62	114	3675	3634.35
100_3	1398.14	2844.66	50.85	109	3913	3692.28
100_4	1682.70	2685.61	37.34	110	3841	3637.24
100_5	1687.09	1861.52	9.37	240	5045	3604.79
250_1	1475.21	2720.64	45.78	113	8427	3639.57
250_2	1504.79	2973.90	49.40	103	8209	3652.97
250_3	835.79	2039.40	59.02	103	7754	3884.88
250_4	1635.15	3640.92	55.09	125	11570	3606.50
250_5	1030.32	2053.56	49.83	114	9040	3620.99
500_1	1260.28	2406.47	47.63	151	12762	3605.09
500_2	1059.06	2452.02	56.81	123	16691	3662.23
500_3	784.44	1637.49	52.09	116	12310	3705.46
500_4	1529.43	2963.04	48.38	74	14035	3602.86
500_5	1173.91	2567.01	54.27	104	15074	3616.05
750_1	1302.92	2616.21	50.20	29	9224	3642.34
750_2	1347.72	2710.59	50.28	41	12576	3716.45
750_4	1364.70	2960.13	53.90	54	16361	3623.96
750_5	1262.64	2470.92	48.90	57	14884	3684.48

Table A.23 presents the results for a 2-periods run, when the model uses only the separation of integer infeasible solutions. Table A.26 is the same, but for a 3-periods run. Table A.24 presents the results where the model uses only the separation of fractional infeasible solutions for a 2-periods run and Table A.27 is the same, but for a 3-periods run. Tables A.25 and A.28 show a summary of these results.

Table A.23: CG instances | Number of periods: 2 | One budget limit throughout all time horizon | Separation of integer infeasible solutions

name	**LB**	**UB**	**gap(%)**	**# nodes**	**# cuts**	**time(s)**
50_1	793.24	793.24	0.00	0	0	0.89
50_2	1323.54	1323.54	0.00	3	0	0.98
50_3	389.08	389.08	0.00	0	0	0.46
50_4	656.58	656.58	0.00	12	0	0.86
50_5	2055.44	2055.44	0.00	89	23	1.82
100_1	1804.10	1804.10	0.00	513	68	11.98
100_2	1737.99	1737.99	0.00	550	83	11.70
100_3	1517.13	1517.17	0.00	537	138	20.18
100_4	1613.58	1613.58	0.00	39	8	4.75
100_5	1351.02	1351.02	0.00	54	22	4.12
250_1	1461.41	1461.41	0.00	698	236	132.13
250_2	1438.05	1438.18	0.01	851	464	170.59
250_3	907.59	907.65	0.01	1166	508	192.02
250_4	1531.68	1531.80	0.01	1065	221	151.51
250_5	1018.10	1018.17	0.01	1564	182	233.56
500_1	1245.34	1245.46	0.01	1254	196	813.52
500_2	1155.87	1155.98	0.01	6615	2697	3582.98
500_3	865.02	865.07	0.01	1801	911	1420.84
500_4	1494.49	1494.64	0.01	2370	806	2408.44
500_5	1151.49	1151.58	0.01	673	266	462.36
750_1	1264.87	1269.27	0.35	2961	1530	3602.71
750_2	1269.76	1282.01	0.96	2270	1054	3600.98
750_3	1403.47	1566.76	10.42	1875	3233	3601.96
750_4	1300.74	1300.87	0.01	1830	283	2220.39
750_5	1217.56	1388.13	12.29	1399	2254	3615.01

Table A.24: CG instances | Number of periods: 2 | One budget limit throughout all time horizon | Separation of fractional infeasible solutions

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	793.24	793.24	0.00	0	0	0.73
50_2	1323.54	1323.54	0.00	11	88	0.75
50_3	389.08	389.08	0.00	0	0	0.39
50_4	656.58	656.58	0.00	3	26	1.20
50_5	2055.44	2055.44	0.00	58	464	207.13
100_1	1804.10	1804.10	0.00	129	992	306.76
100_2	1737.99	1737.99	0.00	241	1585	1692.12
100_3	1409.83	1580.71	10.81	61	1511	3741.52
100_4	1613.58	1613.58	0.00	47	381	169.54
100_5	1351.02	1351.02	0.00	26	107	2.98
250_1	1444.51	1545.37	6.53	41	2745	3782.95
250_2	1430.11	1452.93	1.57	75	3383	3602.83
250_3	904.52	907.65	0.35	134	3084	3632.52
250_4	1513.25	1538.03	1.61	51	1745	3698.57
250_5	983.82	1035.91	5.03	32	2233	3604.24
500_1	1233.18	1257.63	1.94	16	1844	3607.70
500_2	1144.73	1195.02	4.21	25	2434	3744.81
500_3	855.82	889.40	3.78	66	4571	3634.45
500_4	1488.50	1602.11	7.09	12	1562	3606.71
500_5	1147.48	1153.95	0.56	51	2786	3623.90
750_1	1266.37	1363.15	7.10	27	2876	3605.91
750_2	1268.21	1285.43	1.34	15	2282	3625.76
750_3	1407.44	1579.53	10.89	20	3048	3703.23
750_4	1297.56	1301.16	0.28	102	5812	3826.17
750_5	1222.09	1348.03	9.34	12	2191	3616.84

Table A.25: CG instances | Number of periods: 2 | Best results

name	LB	UB	gap(%)	# nodes	# cuts	time(s)	sep.
50_1	793.24	793.24	0.00	0	0	0.47	Both
50_2	1323.54	1323.54	0.00	11	88	1.01	Both
50_3	389.08	389.08	0.00	0	0	0.51	Both
50_4	656.58	656.58	0.00	3	26	0.79	Both

Continued on next page

Table A.25 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)	sep.
50_5	2055.44	2055.44	0.00	67	622	643.06	Both
100_1	1804.10	1804.10	0.00	103	1011	667.86	Both
100_2	1737.99	1737.99	0.00	274	1989	2593.10	Both
100_3	1517.13	1517.17	0.00	537	138	20.18	Int
100_4	1613.58	1613.58	0.00	49	381	189.48	Both
100_5	1351.02	1351.02	0.00	27	132	28.37	Both
250_1	1461.41	1461.41	0.00	698	236	132.13	Int
250_2	1438.05	1438.18	0.01	851	464	170.59	Int
250_3	907.59	907.65	0.01	1166	508	192.02	Int
250_4	1531.68	1531.80	0.01	1065	221	151.51	Int
250_5	1018.10	1018.17	0.01	1564	182	233.56	Int
500_1	1245.34	1245.46	0.01	1254	196	813.52	Int
500_2	1155.87	1155.98	0.01	6615	2697	3582.98	Int
500_3	865.02	865.07	0.01	1801	911	1420.84	Int
500_4	1494.49	1494.64	0.01	2370	806	2408.44	Int
500_5	1151.49	1151.58	0.01	673	266	462.36	Int
750_1	1264.87	1269.27	0.35	2961	1530	3602.71	Int
750_2	1269.76	1282.01	0.96	2270	1054	3600.98	Int
750_3	1409.48	1453.31	3.02	19	2939	3622.45	Both
750_4	1300.74	1300.87	0.01	1830	283	2220.39	Int
750_5	1222.07	1261.19	3.10	17	2351	3716.72	Both

Table A.26: CG instances | Number of periods: 3 | One budget limit throughout all time horizon | Separation of integer infeasible solutions

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	1351.17	1351.17	0.00	31	14	1.81
50_2	1910.28	1910.28	0.00	3	4	1.49
50_3	583.62	583.62	0.00	0	0	0.62
50_4	1139.70	1139.70	0.00	90	14	2.30
50_5	2876.41	2876.41	0.00	570	36	6.03
100_1	2501.09	2501.14	0.00	1221	73	44.88
100_2	2465.02	2465.02	0.00	2249	458	113.84
100_3	2014.80	2014.80	0.00	807	124	44.31
100_4	2385.79	2385.98	0.01	1717	76	64.03
100_5	1861.49	1861.52	0.00	125	6	8.68

Continued on next page

Table A.26 – *Continued from previous page*

name	**LB**	**UB**	**gap(%)**	**# nodes**	**# cuts**	**time(s)**
250_1	1999.73	1999.93	0.01	5096	656	2465.64
250_2	2016.27	2016.46	0.01	6841	585	1939.07
250_3	1253.56	1253.66	0.01	2193	532	1212.95
250_4	2174.81	2177.25	0.11	7090	903	3601.09
250_5	1379.17	1461.06	5.60	4780	763	3601.91
500_1	1717.31	1775.31	3.27	619	353	3604.77
500_2	1549.15	1605.80	3.53	1812	1618	3600.30
500_3	1132.50	1220.35	7.20	1563	836	3601.06
500_4	2003.35	2177.14	7.98	2084	1875	3602.52
500_5	1582.59	1584.88	0.14	3374	961	3600.77
750_1	1688.88	1730.40	2.40	940	1487	3602.42
750_2	1746.58	1892.57	7.71	235	863	3611.61
750_3	1891.95	2197.70	13.91	473	1271	3604.76
750_4	1799.33	1844.56	2.45	1046	1406	3605.02
750_5	1573.84	1810.98	13.09	690	1435	3607.26

Table A.27: CG instances | Number of periods: 3 | One budget limit throughout all time horizon | Separation of fractional infeasible solutions

name	**LB**	**UB**	**gap(%)**	**# nodes**	**# cuts**	**time(s)**
50_1	1351.17	1351.17	0.00	23	81	1.68
50_2	1910.28	1910.28	0.00	3	33	1.05
50_3	583.62	583.62	0.00	0	0	0.55
50_4	1139.70	1139.70	0.00	55	301	72.04
50_5	2876.17	2876.41	0.01	232	1901	3575.92
100_1	2419.27	2606.41	7.18	66	1906	3625.63
100_2	2320.98	2581.15	10.08	80	1985	3633.65
100_3	2109.53	10.26	60	1635	3611.89	
100_4	2212.82	2449.92	9.68	65	1735	3622.89
100_5	1861.48	1861.52	0.00	128	914	866.58
250_1	1964.16	2038.23	3.63	30	2438	3758.28
250_2	1964.60	2043.62	3.87	29	2080	3693.82
250_3	1206.48	1273.62	5.27	78	3030	3862.97
250_4	2121.33	2259.98	6.14	15	1542	3602.76
250_5	1354.71	1490.08	9.08	12	1308	3755.12
500_1	1719.86	1874.27	8.24	6	1600	3611.89

Continued on next page

Table A.27 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
500_2	1546.71	1589.67	2.70	25	2092	3620.40
500_3	1143.29	1200.46	4.76	21	3300	3899.48
500_4	2010.45	2331.80	13.78	9	1797	3611.07
500_5	1571.18	1721.25	8.72	22	1788	3613.12
750_1	1707.24	2090.23	18.32	9	1655	3876.94
750_2	1779.08	2066.99	13.93	7	2017	3844.28
750_3	1918.00	2140.98	10.42	8	2600	3643.16
750_4	1819.70	2049.52	11.21	9	1905	3906.80
750_5	1602.25	1859.45	13.83	12	2384	3636.34

Table A.28: CG instances | Number of periods: 3 | Best results

name	LB	UB	gap(%)	# nodes	# cuts	time(s)	sep.
50_1	1351.17	1351.17	0.00	23	90	1.77	Both
50_2	1910.28	1910.28	0.00	3	54	1.29	Both
50_3	583.62	583.62	0.00	0	0	0.69	Both
50_4	1139.70	1139.70	0.00	55	301	72.16	Both
50_5	2876.41	2876.41	0.00	228	1871	2713.97	Both
100_1	2501.09	2501.14	0.00	1221	73	44.88	Int
100_2	2465.02	2465.02	0.00	2249	458	113.84	Int
100_3	2014.80	2014.80	0.00	807	124	44.31	Int
100_4	2385.79	2385.98	0.01	1717	76	64.03	Int
100_5	1861.48	1861.52	0.00	119	848	670.15	Both
250_1	1999.73	1999.93	0.01	5096	656	2465.64	Int
250_2	2016.27	2016.46	0.01	6841	585	1939.07	Int
250_3	1253.56	1253.66	0.01	2193	532	1212.95	Int
250_4	2174.81	2177.25	0.11	7090	903	3601.09	Int
250_5	1379.17	1461.06	5.60	4780	763	3601.91	Int
500_1	1717.31	1775.31	3.27	619	353	3604.77	Int
500_2	1546.71	1589.67	2.70	25	2092	3620.40	Frac
500_3	1143.46	1200.46	4.75	23	3858	3761.12	Both
500_4	2003.35	2177.14	7.98	2084	1875	3602.52	Int
500_5	1582.59	1584.88	0.14	3374	961	3600.77	Int
750_1	1688.88	1730.40	2.40	940	1487	3602.42	Int
750_2	1746.58	1892.57	7.71	235	863	3611.61	Int
750_3	1918.00	2140.98	10.42	9	2600	3632.77	Both

Continued on next page

Table A.28 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)	sep.
750_4	1799.33	1844.56	2.45	1046	1406	3605.02	Int
750_5	1573.84	1810.98	13.09	690	1435	3607.26	Int

Tables A.29 and A.30 show the results for our model and Suhl and Hilbert's, respectively, for a 2-period run and, finally, Tables A.31 and A.32 show the results for a 5-periods run.

Table A.29: CG instances | Number of periods: 2 | One budget limit per period

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	793.24	793.24	0.00	0	0	0.78
50_2	1323.54	1323.54	0.00	5	38	0.89
50_3	389.08	389.08	0.00	0	0	0.48
50_4	656.58	656.58	0.00	3	21	0.77
50_5	2055.44	2055.44	0.00	75	593	593.46
100_1	1804.10	1804.10	0.00	101	826	471.43
100_2	1688.55	1788.62	5.59	159	2353	3660.83
100_3	1389.22	1576.54	11.88	69	1730	3634.66
100_4	1613.58	1613.58	0.00	49	479	454.73
100_5	1351.02	1351.02	0.00	23	182	3.21
250_1	1444.49	1477.78	2.25	81	2959	3681.45
250_2	1426.05	1467.90	2.85	50	2405	3604.39
250_3	897.21	961.39	6.68	70	2407	3625.48
250_4	1516.84	1537.76	1.36	106	2233	3606.62
250_5	974.52	1026.20	5.04	73	2379	3603.26
500_1	1238.41	1257.63	1.53	30	2574	3642.08
500_2	1145.38	1195.02	4.15	20	2240	3771.07
500_3	855.23	1091.66	21.66	20	2994	3607.94
500_4	1484.16	1504.24	1.33	58	3395	3694.55
500_5	1147.49	1159.34	1.02	53	2324	3905.51
750_1	1258.88	1363.15	7.65	10	1363	3606.42
750_2	1269.34	1358.69	6.58	13	2637	3893.03
750_3	1409.19	1566.54	10.04	12	2619	3908.51
750_4	1297.55	1303.43	0.45	70	5034	3630.16
750_5	1220.96	1235.24	1.16	22	2885	3851.46

Table A.30: CG instances | Number of periods: 2 | One budget limit per period | SUHL AND HILBERT

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	793.24	793.24	0.00	23	0	0.49
50_2	1323.54	1323.54	0.00	72	0	0.60
50_3	389.08	389.08	0.00	0	0	0.43
50_4	656.58	656.58	0.00	89	3	0.91
50_5	2055.42	2055.44	0.00	7834	33	17.88
100_1	1803.99	1804.10	0.01	35023	76	287.55
100_2	1737.94	1737.99	0.00	33627	46	275.73
100_3	1517.15	1517.17	0.00	38704	43	310.66
100_4	1613.55	1613.58	0.00	7305	36	70.07
100_5	1351.02	1351.02	0.00	2156	8	18.88
250_1	1302.55	1466.45	11.18	84501	59	3601.18
250_2	1283.88	1442.47	10.99	77501	105	3603.75
250_3	794.65	907.65	12.45	86801	34	3600.50
250_4	1363.11	1535.60	11.23	78634	68	3600.16
250_5	860.17	1021.53	15.80	80930	46	3600.16
500_1	1048.67	1270.66	17.47	6093	47	3600.43
500_2	972.48	1171.21	16.97	5301	53	3618.31
500_3	742.35	880.33	15.67	9201	52	3622.05
500_4	1304.51	1932.11	32.48	5801	81	3605.97
500_5	1027.46	-	-	2682	58	3602.73
750_1	1071.02	-	-	920	13	3687.58
750_2	1075.66	-	-	460	11	3622.17
750_3	1221.51	-	-	570	18	3797.66
750_4	1085.04	-	-	412	9	3600.53
750_5	1024.01	-	-	650	20	3694.92

Table A.31: CG instances | Number of periods: 5 | One budget limit per period

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	2133.01	2133.01	0.00	17	47	2.51
50_2	3268.79	3268.79	0.00	19	333	129.96
50_3	972.70	972.70	0.00	5	40	11.57
50_4	1899.50	1899.50	0.00	19	135	7.50
50_5	4077.11	4571.41	10.81	152	2678	3619.91

Table A.32: CG instances | Number of periods: 5 | One budget limit per period | SUHL AND HILBERT

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	2133.01	2133.01	0.00	206	6	2.15
50_2	3268.79	3268.79	0.00	1306	9	6.90
50_3	972.70	972.70	0.00	22	0	0.72
50_4	1899.50	1899.50	0.00	1364	4	5.68
50_5	4361.35	4361.64	0.01	41300	15	287.22

A.3
Randomly generated instances of incomplete graphs

At first, we will present in Tables A.33, A.34, A.35, A.36 and A.37 the results for all five instances of each instance size set, considering all connectivity constraints and all separation procedures, for a 3-periods, 5-periods, 8-periods, 10-periods and 15-periods runs, respectively.

Table A.33: IG instances | Number of periods: 3 | One budget limit throughout all time horizon

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	22.29	22.29	0.00	0	0	0.25
50_2	22.59	22.59	0.00	3	10	0.29
50_3	21.75	21.75	0.00	4	41	0.41
50_4	22.26	22.26	0.00	0	0	0.23
50_5	11.86	11.86	0.00	0	6	0.33
100_1	27.84	27.84	0.00	34	453	7.07
100_2	33.18	33.18	0.00	55	797	244.60
100_3	28.31	28.31	0.00	36	541	27.35
100_4	32.97	32.97	0.00	42	692	9.25
100_5	23.09	23.09	0.00	19	127	10.94
150_1	29.75	29.75	0.00	100	1568	237.65
150_2	34.48	34.48	0.00	88	1717	539.84
150_3	39.03	39.03	0.00	97	2750	574.89
150_4	39.28	39.28	0.00	133	2026	446.35
150_5	37.73	37.73	0.00	135	2289	1659.27
200_1	36.59	48.47	24.51	60	5123	3780.00
200_2	32.59	41.44	21.36	120	8348	3845.49

Continued on next page

Table A.33 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
200_3	31.07	45.43	31.60	67	3500	3682.42
200_4	29.04	43.14	32.68	107	4707	3600.41
200_5	34.59	46.12	24.99	77	4237	3612.53

Table A.34: IG instances | Number of periods: 5 | One budget limit throughout all time horizon

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	37.15	37.15	0.00	0	0	0.79
50_2	37.65	37.65	0.00	8	16	0.70
50_3	36.25	36.25	0.00	9	44	1.10
50_4	37.10	37.10	0.00	0	0	0.38
50_5	19.70	19.70	0.00	3	19	1.04
100_1	44.20	44.20	0.00	37	787	21.17
100_2	51.09	51.09	0.00	100	1623	762.57
100_3	44.67	44.67	0.00	55	1028	162.32
100_4	53.39	53.39	0.00	101	1452	239.11
100_5	33.91	33.91	0.00	19	234	1.84
150_1	43.58	43.58	0.00	236	5441	3185.35
150_2	57.32	57.32	0.00	133	4523	2758.80
150_3	47.09	64.80	27.34	65	3782	3732.80
150_4	53.28	59.13	9.89	184	5165	3610.09
150_5	45.41	67.22	32.44	94	3997	3600.45

Table A.35: IG instances | Number of periods: 8 | One budget limit throughout all time horizon

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	59.44	59.44	0.00	0	0	0.45
50_2	60.24	60.24	0.00	13	22	0.71
50_3	58.00	58.00	0.00	14	79	1.31
50_4	59.36	59.36	0.00	0	0	0.37
50_5	31.46	31.46	0.00	5	17	0.64
100_1	68.74	68.74	0.00	57	1212	74.82
100_2	75.96	75.96	0.00	234	3666	1052.68

Continued on next page

Table A.35 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
100_3	69.21	69.21	0.00	83	1722	219.59
100_4	78.11	78.11	0.00	192	3505	981.04
100_5	50.14	50.14	0.00	28	278	38.92

Table A.36: IG instances | Number of periods: 10 | One budget limit throughout all time horizon

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_5	39.30	39.30	0.00	5	22	0.79
50_1	74.30	74.30	0.00	0	0	0.38
50_2	75.30	75.30	0.00	15	46	0.81
50_3	72.50	72.50	0.00	22	129	1.99
50_4	74.20	74.20	0.00	0	0	0.40
50_5	39.30	39.30	0.00	5	22	0.73
100_1	85.10	85.10	0.00	103	1868	415.74
100_2	88.49	88.49	0.00	264	3551	987.62
100_3	85.57	85.57	0.00	63	1559	268.03
100_4	94.59	94.59	0.00	128	3074	543.92
100_5	60.96	60.96	0.00	35	425	20.37

Table A.37: IG instances | Number of periods: 15 | One budget limit throughout all time horizon

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	111.45	111.45	0.00	0	0	0.81
50_2	112.95	112.95	0.00	6	63	1.64
50_3	108.75	108.75	0.00	45	168	4.65
50_4	111.30	111.30	0.00	0	0	0.89
50_5	58.90	58.90	0.00	9	37	2.42
100_1	126.00	126.00	0.00	149	2501	778.26
100_2	116.23	116.24	0.01	542	7142	2094.73
100_3	126.47	126.47	0.00	155	2977	487.09
100_4	135.79	135.79	0.00	274	4726	1111.38
100_5	88.01	88.01	0.00	66	893	72.26

Tables A.38 and A.39 show results of runs without valid inequalities, for 2 periods and 3 periods, respectively.

Table A.38: IG instances | Number of periods: 2 | One budget limit throughout all time horizon | Without connectivity constraints

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	14.86	14.86	0.00	0	0	0.33
50_2	11.91	11.91	0.00	41	244	0.79
50_3	14.50	14.50	0.00	11	142	0.63
50_4	14.84	14.84	0.00	55	342	10.81
50_5	4.84	4.84	0.00	23	176	1.21
100_1	16.68	16.68	0.00	129	2837	188.65
100_2	20.04	20.04	0.00	165	1970	497.06
100_3	16.95	16.95	0.00	60	891	7.11
100_4	19.91	19.91	0.00	258	4326	2541.37
100_5	11.83	11.83	0.00	42	409	1.60
150_1	16.60	16.60	0.00	194	2523	573.45
150_2	21.87	22.91	4.52	248	7884	3658.73
150_3	21.99	25.32	13.16	232	6467	3829.56
150_4	22.74	22.74	0.00	301	4625	2270.83
150_5	17.15	26.96	36.38	198	3917	3616.09
200_1	31.12	31.12	0.00	340	8407	1995.89
200_2	26.83	26.83	0.00	196	2865	27.84
200_3	26.13	31.21	16.28	279	9344	3755.44
200_4	28.23	28.23	0.00	314	7881	1611.55
200_5	31.02	31.02	0.00	422	9852	2904.94
250_1	31.53	35.47	11.12	225	10096	3716.27
250_2	28.81	34.65	16.86	254	12770	3609.12
250_3	27.83	40.93	32.01	170	9224	3620.86
250_4	31.31	32.55	3.82	689	10705	3694.09
250_5	33.12	36.30	8.76	406	15681	3613.94
300_1	21.58	39.83	45.82	157	11825	3884.84
300_2	30.55	49.36	38.10	223	10284	3609.36
300_3	32.67	55.08	40.69	174	11746	3600.72
300_4	30.15	45.25	33.37	135	10717	3775.95
300_5	30.56	51.64	40.82	159	7542	3610.98

Table A.39: IG instances | Number of periods: 3 | One budget limit throughout all time horizon | Without connectivity constraints

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	22.29	22.29	0.00	0	0	0.82
50_2	22.59	22.59	0.00	11	180	0.64
50_3	21.75	21.75	0.00	27	229	0.97
50_4	22.26	22.26	0.00	4	166	0.70
50_5	11.86	11.86	0.00	3	178	1.33
100_1	27.84	27.84	0.00	73	1015	2.59
100_2	33.18	33.18	0.00	175	1915	54.72
100_3	28.31	28.31	0.00	80	1100	7.07
100_4	32.97	32.97	0.00	182	1976	119.87
100_5	23.09	23.09	0.00	23	427	1.27
150_1	29.75	29.75	0.00	196	3237	381.17
150_2	34.48	34.48	0.00	123	2547	84.18
150_3	27.26	40.74	33.09	226	7800	3762.02
150_4	39.28	39.28	0.00	378	5996	1535.04
150_5	31.94	38.74	17.56	234	5903	3639.07
200_1	27.79	52.19	46.75	129	11882	3701.61
200_2	27.17	47.29	42.54	197	8982	3664.12
200_3	28.19	51.12	44.86	141	8595	3607.87
200_4	21.97	45.05	51.24	165	9748	3611.84
200_5	25.29	47.96	47.26	145	7853	3605.75

Table A.40 shows the results for a 2-periods run, when the model uses only the separation of integer infeasible solutions. Table A.43 is the same, but for a 3-periods run. Table A.41 shows the results where the model uses only the separation of fractional infeasible solutions for a 2-periods run and Table A.44 is the same, but for a 3-periods run. Tables A.42 and A.45 show a summary of these results.

Table A.40: IG instances | Number of periods: 2 | Separation of integer infeasible solutions

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	14.86	14.86	0.00	0	0	0.71
50_2	11.91	11.91	0.00	13	4	0.54
50_3	14.50	14.50	0.00	0	6	0.33

Continued on next page

Table A.40 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_4	14.84	14.84	0.00	55	0	0.55
50_5	4.84	4.84	0.00	9	10	0.54
100_1	16.68	16.68	0.00	281	22	2.89
100_2	20.04	20.04	0.00	1118	48	5.88
100_3	16.95	16.95	0.00	511	48	4.42
100_4	19.91	19.91	0.00	1678	72	8.36
100_5	11.83	11.83	0.00	23	6	1.35
150_1	16.60	16.60	0.00	1413	181	10.81
150_2	22.91	22.91	0.00	5085	314	44.34
150_3	25.32	25.32	0.00	18512	407	223.35
150_4	22.74	22.74	0.00	4945	181	31.72
150_5	24.23	24.23	0.00	4561	134	31.08
200_1	31.12	31.12	0.00	7443	423	80.06
200_2	26.83	26.83	0.00	812	150	10.56
200_3	28.97	28.97	0.00	5215	418	53.45
200_4	28.23	28.23	0.00	3744	235	39.35
200_5	31.02	31.02	0.00	10417	466	116.23
250_1	35.26	35.26	0.00	47609	682	861.10
250_2	33.73	33.73	0.00	20483	905	260.46
250_3	35.62	35.62	0.00	45877	618	736.77
250_4	32.54	32.54	0.00	6306	418	73.46
250_5	36.19	36.19	0.00	34433	288	439.33
300_1	33.73	37.80	10.76	71460	2170	3600.09
300_2	35.72	44.44	19.61	52201	2736	3600.28
300_3	42.53	42.53	0.00	95217	1520	2264.45
300_4	38.60	44.44	13.15	52911	1622	3600.90
300_5	40.70	47.30	13.96	46800	2526	3600.98

Table A.41: IG instances | Number of periods: 2 | Separation of fractional infeasible solutions

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	14.86	14.86	0.00	0	0	0.21
50_2	11.91	11.91	0.00	9	49	10.33
50_3	14.50	14.50	0.00	7	7	0.30
50_4	14.84	14.84	0.00	17	265	36.04

Continued on next page

Table A.41 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_5	4.84	4.84	0.00	14	32	0.36
100_1	16.68	16.68	0.00	49	928	89.89
100_2	20.04	20.04	0.00	64	1133	310.38
100_3	16.95	16.95	0.00	67	1829	659.46
100_4	19.91	19.91	0.00	59	1148	730.68
100_5	11.83	11.83	0.00	11	149	0.82
150_1	16.60	16.60	0.00	66	975	309.74
150_2	20.50	22.97	10.74	125	3920	3778.09
150_3	23.01	25.35	9.23	201	5210	3776.65
150_4	22.74	22.74	0.00	195	2294	892.56
150_5	24.23	24.23	0.00	144	3032	1619.47
200_1	31.12	31.12	0.00	202	4936	2309.26
200_2	26.83	26.83	0.00	78	2302	158.00
200_3	28.97	28.97	0.00	81	2422	1069.68
200_4	28.23	28.23	0.00	111	3456	1341.32
200_5	31.02	31.02	0.00	141	4009	1281.45
250_1	33.06	35.47	6.81	108	7525	3747.05
250_2	30.93	33.99	9.01	192	6020	3607.32
250_3	34.32	35.62	3.64	179	5753	3681.54
250_4	29.54	34.15	13.49	445	6332	3619.88
250_5	33.41	36.20	7.70	185	8348	3680.47
300_1	32.52	43.41	25.09	65	5682	3745.85
300_2	36.10	45.52	20.70	42	3678	3615.59
300_3	38.72	51.97	25.50	90	17174	3615.49
300_4	40.87	42.95	4.84	52	8150	3686.24
300_5	41.12	51.60	20.32	78	12888	3675.16

Table A.42: IG instances | Number of periods: 2 | Best results

name	LB	UB	gap(%)	# nodes	# cuts	time(s)	sep.
50_1	14.86	14.86	0.00	0	0	0.25	Both
50_2	11.91	11.91	0.00	9	60	20.50	Both
50_3	14.50	14.50	0.00	7	7	0.39	Both
50_4	14.84	14.84	0.00	17	265	36.11	Both
50_5	4.84	4.84	0.00	14	37	0.56	Both
100_1	16.68	16.68	0.00	25	564	84.69	Both

Continued on next page

Table A.42 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)	sep.
100_2	20.04	20.04	0.00	71	1059	326.21	Both
100_3	16.95	16.95	0.00	37	1011	47.38	Both
100_4	19.91	19.91	0.00	84	1525	1054.65	Both
100_5	11.83	11.83	0.00	13	185	1.04	Both
150_1	16.60	16.60	0.00	99	1026	356.97	Both
150_2	22.91	22.91	0.00	143	4425	2464.07	Both
150_3	25.32	25.32	0.00	18512	407	223.35	Int
150_4	22.74	22.74	0.00	179	2653	1676.76	Both
150_5	24.23	24.23	0.00	4561	134	31.08	Int
200_1	31.12	31.12	0.00	7443	423	80.06	Int
200_2	26.83	26.83	0.00	106	3792	1095.22	Both
200_3	28.97	28.97	0.00	104	2485	682.51	Both
200_4	28.23	28.23	0.00	3744	235	39.35	Int
200_5	31.02	31.02	0.00	202	5046	1956.98	Both
250_1	35.26	35.26	0.00	47609	682	861.10	Int
250_2	33.73	33.73	0.00	20483	905	260.46	Int
250_3	35.62	35.62	0.00	45877	618	736.77	Int
250_4	32.54	32.54	0.00	6306	418	73.46	Int
250_5	36.19	36.19	0.00	34433	288	439.33	Int
300_1	33.73	37.80	10.76	71460	2170	3600.09	Int
300_2	35.72	44.44	19.61	52201	2736	3600.28	Int
300_3	42.53	42.53	0.00	95217	1520	2264.45	Int
300_4	40.87	42.95	4.84	52	8150	3686.24	Frac
300_5	40.70	47.30	13.96	46800	2526	3600.98	Int

Table A.43: IG instances | Number of periods: 3 | Separation of integer infeasible solutions

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	22.29	22.29	0.00	0	0	0.30
50_2	22.59	22.59	0.00	0	0	0.28
50_3	21.75	21.75	0.00	10	9	0.60
50_4	22.26	22.26	0.00	0	0	0.49
50_5	11.86	11.86	0.00	0	0	0.40
100_1	27.84	27.84	0.00	128	18	2.25
100_2	33.18	33.18	0.00	337	35	3.77

Continued on next page

Table A.43 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
100_3	28.31	28.31	0.00	110	10	2.47
100_4	32.97	32.97	0.00	662	32	4.01
100_5	23.09	23.09	0.00	9	0	0.95
150_1	29.75	29.75	0.00	1115	30	8.39
150_2	34.48	34.48	0.00	503	45	6.15
150_3	39.03	39.03	0.00	3208	44	32.89
150_4	39.28	39.28	0.00	1367	68	14.83
150_5	37.73	37.73	0.00	2509	83	17.03
200_1	45.05	45.05	0.00	106441	1004	3471.20
200_2	36.87	36.87	0.00	5676	323	98.95
200_3	41.38	41.38	0.00	60862	1010	1360.33
200_4	41.15	41.15	0.00	71369	1602	1773.56
200_5	43.99	43.99	0.00	85100	1083	3251.98

Table A.44: IG instances | Number of periods: 3 | Separation of fractional infeasible solutions

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	22.29	22.29	0.00	0	0	0.22
50_2	22.59	22.59	0.00	3	8	0.27
50_3	21.75	21.75	0.00	9	27	0.42
50_4	22.26	22.26	0.00	0	0	0.19
50_5	11.86	11.86	0.00	0	6	0.26
100_1	27.84	27.84	0.00	24	301	6.53
100_2	33.18	33.18	0.00	46	599	58.94
100_3	28.31	28.31	0.00	39	554	57.30
100_4	32.97	32.97	0.00	48	736	23.12
100_5	23.09	23.09	0.00	15	127	5.90
150_1	29.75	29.75	0.00	133	1581	345.88
150_2	34.48	34.48	0.00	65	1553	261.19
150_3	39.03	39.03	0.00	116	2907	757.33
150_4	39.28	39.28	0.00	131	1928	597.67
150_5	37.73	37.73	0.00	106	1667	1135.18
200_1	37.69	45.47	17.10	117	7757	3675.15
200_2	33.37	40.78	18.18	90	6069	3612.55
200_3	33.87	53.55	36.75	40	4000	3693.14

Continued on next page

Table A.44 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
200_4	28.48	41.97	32.13	60	3575	3627.45
200_5	30.77	44.51	30.88	75	6179	3705.55

Table A.45: IG instances | Number of periods: 3 | Best results

name	LB	UB	gap(%)	# nodes	# cuts	time(s)	sep.
50_1	22.29	22.29	0.00	0	0	0.25	Both
50_2	22.59	22.59	0.00	3	10	0.29	Both
50_3	21.75	21.75	0.00	4	41	0.41	Both
50_4	22.26	22.26	0.00	0	0	0.23	Both
50_5	11.86	11.86	0.00	0	6	0.33	Both
100_1	27.84	27.84	0.00	34	453	7.07	Both
100_2	33.18	33.18	0.00	55	797	244.60	Both
100_3	28.31	28.31	0.00	36	541	27.35	Both
100_4	32.97	32.97	0.00	42	692	9.25	Both
100_5	23.09	23.09	0.00	19	127	10.94	Both
150_1	29.75	29.75	0.00	100	1568	237.65	Both
150_2	34.48	34.48	0.00	88	1717	539.84	Both
150_3	39.03	39.03	0.00	97	2750	574.89	Both
150_4	39.28	39.28	0.00	133	2026	446.35	Both
150_5	37.73	37.73	0.00	135	2289	1659.27	Both
200_1	45.05	45.05	0.00	106441	1004	3471.20	Int
200_2	36.87	36.87	0.00	5676	323	98.95	Int
200_3	41.38	41.38	0.00	60862	1010	1360.33	Int
200_4	41.15	41.15	0.00	71369	1602	1773.56	Int
200_5	43.99	43.99	0.00	85100	1083	3251.98	Int

Tables A.46 and A.47 show the results for our model and Suhl and Hilbert's, respectively, for a 1-period run. Tables A.48 and A.49 do the same comparison for a 3-periods run and, finally, Tables A.50 and A.51 act similarly for a 5-periods run.

Table A.46: IG instances | Number of periods: 1 | One budget limit per period

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	7.43	7.43	0.00	0	0	0.21

Continued on next page

Table A.46 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_2	4.37	4.37	0.00	2	2	0.23
50_3	7.25	7.25	0.00	3	16	0.28
50_4	7.42	7.42	0.00	9	62	0.34
50_5	0.92	0.92	0.00	0	3	0.24
100_1	8.50	8.50	0.00	25	369	16.02
100_2	8.98	8.98	0.00	15	181	0.90
100_3	8.77	8.77	0.00	26	335	11.11
100_4	8.96	8.96	0.00	36	289	5.98
100_5	6.42	6.42	0.00	5	31	0.50
150_1	7.74	7.74	0.00	47	350	151.64
150_2	11.56	11.56	0.00	65	888	58.35
150_3	11.72	11.72	0.00	27	448	2.03
150_4	11.24	11.24	0.00	18	201	21.11
150_5	11.78	11.78	0.00	39	498	82.46
200_1	15.18	15.18	0.00	41	899	39.87
200_2	12.69	12.69	0.00	29	377	7.43
200_3	14.63	14.63	0.00	94	2214	689.61
200_4	14.32	14.32	0.00	159	3516	434.24
200_5	14.99	14.99	0.00	131	3758	820.50
250_1	15.92	17.96	11.36	112	4789	3679.14
250_2	16.25	16.25	0.00	134	1460	525.43
250_3	16.22	16.22	0.00	101	4737	2836.20
250_4	14.64	14.64	0.00	208	2264	396.86
250_5	16.69	16.69	0.00	159	4377	642.75
300_1	17.65	18.42	4.18	118	4409	3854.41
300_2	17.55	18.69	6.08	97	3746	3792.08
300_3	20.91	20.91	0.00	43	1187	61.86
300_4	20.07	24.99	19.68	109	10628	3602.67
300_5	22.34	22.34	0.00	151	6259	3025.37

Table A.47: IG instances | Number of periods: 1 | One budget limit per period | SUHL AND HILBERT

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	7.43	7.43	0.00	0	0	0.36
50_2	4.37	4.37	0.00	7	1	0.12

Continued on next page

Table A.47 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_3	7.25	7.25	0.00	53	2	0.17
50_4	7.42	7.42	0.00	31	0	0.13
50_5	0.92	0.92	0.00	18	5	0.14
100_1	8.50	8.50	0.00	1640	79	1.36
100_2	8.98	8.98	0.00	2644	106	1.94
100_3	8.77	8.77	0.00	2568	33	2.30
100_4	8.96	8.96	0.00	3899	104	3.14
100_5	6.42	6.42	0.00	120	3	0.23
150_1	7.74	7.74	0.00	5790	62	4.89
150_2	11.56	11.56	0.00	29024	271	28.87
150_3	11.72	11.72	0.00	23749	99	24.75
150_4	11.24	11.24	0.00	23982	139	19.95
150_5	11.78	11.78	0.00	19732	139	17.88
200_1	15.18	15.18	0.00	370805	394	391.28
200_2	12.69	12.69	0.00	13639	36	11.97
200_3	14.63	14.63	0.00	249440	500	291.91
200_4	14.32	14.32	0.00	451754	884	483.47
200_5	14.99	14.99	0.00	606116	1455	638.54
250_1	14.85	17.92	17.13	1599595	5181	3600.02
250_2	16.25	16.25	0.00	2219668	1771	3565.09
250_3	16.22	16.22	0.00	905247	357	1163.91
250_4	14.64	14.64	0.00	534985	247	565.55
250_5	16.69	16.69	0.00	881037	425	1171.17
300_1	14.82	18.37	19.31	2088601	767	3600.15
300_2	14.49	18.74	22.70	1727533	1410	3600.05
300_3	18.11	20.91	13.41	2123401	999	3600.21
300_4	17.13	23.00	25.50	1805200	1019	3600.38
300_5	18.63	23.87	21.93	1657100	1084	3600.39

Table A.48: IG instances | Number of periods: 3 | One budget limit per period

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	22.29	22.29	0.00	0	0	0.26
50_2	22.59	22.59	0.00	2	18	0.32
50_3	21.75	21.75	0.00	7	47	0.45
50_4	22.26	22.26	0.00	0	0	0.33

Continued on next page

Table A.48 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_5	11.86	11.86	0.00	0	6	0.38
100_1	27.84	27.84	0.00	28	347	6.90
100_2	33.18	33.18	0.00	55	706	219.15
100_3	28.31	28.31	0.00	40	563	42.77
100_4	32.97	32.97	0.00	39	551	173.35
100_5	23.09	23.09	0.00	21	184	0.99
150_1	29.75	29.75	0.00	119	1772	154.65
150_2	34.48	34.48	0.00	61	1226	279.20
150_3	39.03	39.03	0.00	138	3361	1300.19
150_4	39.28	39.28	0.00	129	1792	768.23
150_5	37.73	37.73	0.00	71	1299	983.32
200_1	32.85	48.47	32.22	38	3704	3934.96
200_2	31.78	42.98	26.06	105	6163	3603.50
200_3	33.22	43.54	23.70	48	3860	3668.90
200_4	30.63	41.20	25.66	77	4821	3618.03
200_5	35.26	47.65	26.01	93	5645	3822.53

Table A.49: IG instances | Number of periods: 3 | One budget limit per period | SUHL AND HILBERT

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
50_1	22.29	22.29	0.00	0	0	0.12
50_2	22.59	22.59	0.00	26	0	0.16
50_3	21.75	21.75	0.00	86	6	0.27
50_4	22.26	22.26	0.00	0	0	0.17
50_5	11.86	11.86	0.00	25	0	0.16
100_1	27.84	27.84	0.00	1107	4	1.35
100_2	33.18	33.18	0.00	4660	26	4.69
100_3	28.31	28.31	0.00	1150	5	1.70
100_4	32.97	32.97	0.00	3096	8	3.61
100_5	23.09	23.09	0.00	14	0	0.32
150_1	29.75	29.75	0.00	6638	12	15.09
150_2	34.48	34.48	0.00	10837	20	21.90
150_3	39.03	39.03	0.00	38178	14	77.28
150_4	39.28	39.28	0.00	42569	55	61.89
150_5	37.73	37.73	0.00	18316	8	30.06

Continued on next page

Table A.49 – *Continued from previous page*

name	**LB**	**UB**	**gap(%)**	**# nodes**	**# cuts**	**time(s)**
200_1	33.70	45.05	25.18	835826	343	3600.17
200_2	36.87	36.87	0.00	345645	50	1652.71
200_3	32.37	44.50	27.25	769900	410	3600.07
200_4	29.09	41.92	30.60	743421	294	3600.01
200_5	31.53	47.38	33.45	917830	433	3600.02

Table A.50: IG instances | Number of periods: 5 | One budget limit per period

name	**LB**	**UB**	**gap(%)**	**# nodes**	**# cuts**	**time(s)**
50_1	37.15	37.15	0.00	0	0	0.58
50_2	37.65	37.65	0.00	4	20	0.74
50_3	36.25	36.25	0.00	9	93	6.15
50_4	37.10	37.10	0.00	0	0	0.63
50_5	19.70	19.70	0.00	3	9	0.72
100_1	44.20	44.20	0.00	39	702	65.12
100_2	51.09	51.09	0.00	73	1353	549.45
100_3	44.67	44.67	0.00	37	781	133.18
100_4	53.39	53.39	0.00	116	1837	441.83
100_5	33.91	33.91	0.00	21	258	41.79
150_1	43.58	43.58	0.00	212	4659	2427.87
150_2	57.32	57.32	0.00	124	5298	2217.67
150_3	45.22	64.80	30.21	89	4558	3619.54
150_4	46.00	61.36	25.03	134	5032	3648.39
150_5	43.62	62.68	30.40	39	1704	3629.60

Table A.51: IG instances | Number of periods: 5 | One budget limit per period | SUHL AND HILBERT

name	**LB**	**UB**	**gap(%)**	**# nodes**	**# cuts**	**time(s)**
50_1	37.15	37.15	0.00	0	0	0.16
50_2	37.65	37.65	0.00	65	0	0.22
50_3	36.25	36.25	0.00	750	10	1.36
50_4	37.10	37.10	0.00	12	0	0.22
50_5	19.70	19.70	0.00	43	0	0.23
100_1	44.20	44.20	0.00	7890	8	10.62

Continued on next page

Table A.51 – *Continued from previous page*

name	LB	UB	gap(%)	# nodes	# cuts	time(s)
100_2	51.09	51.09	0.00	29383	41	39.85
100_3	44.67	44.67	0.00	15526	7	24.88
100_4	53.39	53.39	0.00	41832	16	54.91
100_5	33.91	33.91	0.00	326	0	0.91
150_1	43.58	43.58	0.00	207785	18	791.94
150_2	57.32	57.32	0.00	643497	62	2493.49
150_3	57.76	61.69	6.38	931774	19	3600.07
150_4	59.13	59.13	0.00	604739	97	1889.99
150_5	58.53	58.53	0.00	414171	18	1374.41