UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE CIÊNCIAS

DEPARTAMENTO DE COMPUTAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ROMMEL DIAS SARAIVA

MATHEMATICAL PROGRAMMING APPROACHES FOR NP-HARD CONSTRAINED SHORTEST PATH PROBLEMS

FORTALEZA

2019

ROMMEL DIAS SARAIVA

MATHEMATICAL PROGRAMMING APPROACHES FOR NP-HARD CONSTRAINED SHORTEST PATH PROBLEMS

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de doutor em Ciência da Computação. Área de Concentração: Algoritmos e Otimização

Orientador: Prof. Dr. Rafael Castro de Andrade

FORTALEZA

2019

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S247m Saraiva, Rommel Dias.
Mathematical programming approaches for NP-Hard constrained shortest path problems / Rommel Dias Saraiva. – 2019.
74 f. : il.

Tese (doutorado) – Universidade Federal do Ceará, Centro de Ciências, Programa de Pós-Graduação em Ciência da Computação , Fortaleza, 2019.
Orientação: Prof. Dr. Rafael Castro de Andrade.

1. Combinatorial optimization. 2. Shortest path with negative cycles. 3. Constrained shortest path tour problem. 4. Integer linear programming. 5. Lagrangian relaxation. I. Título.

CDD 005

ROMMEL DIAS SARAIVA

MATHEMATICAL PROGRAMMING APPROACHES FOR NP-HARD CONSTRAINED SHORTEST PATH PROBLEMS

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de doutor em Ciência da Computação. Área de Concentração: Algoritmos e Otimização

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. Rafael Castro de Andrade (Orientador)
Universidade Federal do Ceará (UFC)

Profa. Dra. Ana Karolinna Maia de Oliveira
Universidade Federal do Ceará (UFC)

Prof. Dr. Manoel Bezerra Campêlo Neto
Universidade Federal do Ceará (UFC)

Prof. Dr. Marcos José Negreiros Gomes
Universidade Estadual do Ceará (UECE)

Prof. Dr. Plácido Rogério Pinheiro
Universidade de Fortaleza (UNIFOR)

Prof. Dr. Nelson Maculan Filho
Universidade Federal do Rio de Janeiro (UFRJ)

To my family.

ACKNOWLEDGEMENTS

This thesis is fruit of a research developed in the last four and half years. During this time, there existed some people and entities that, directly or indirectly, were important for the accomplishment of this work, and to whom I would like to express my gratitude.

First and foremost, I would like to thank my advisor Prof. Rafael Andrade for believing in me, for giving me opportunity to do this research, and for guiding me during the PhD program with his constant support and advices.

I would like to thank the members of the examination board – Profa. Ana Karolinna, Prof. Manoel Campêlo, Prof. Marcos Negreiros, Prof. Plácido Pinheiro and Prof. Nelson Maculan – for their presence and constructive comments on the work.

I would like to thank the ParGO research group. To all faculty members, for the knowledge transmitted during lectures. To all laboratory colleagues, for the companionship. A particular thank goes to Tatiane, Mardson, Ernando, Jefferson, Adriano and Sérgio. Talking about issues related to research with them was always of great importance. Our friendship outside the laboratory was also essential to take a break in moments of research stress.

I would like to thank FUNCAP (Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico) and CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) for the financial support.

Lastly, but not least, I would like to thank my family for their support and love they provided me during my entire life.

RESUMO

Neste trabalho, estudamos dois problemas de roteamento NP-Difíceis: o problema do caminho mínimo na presença de ciclos negativos (referenciado na literatura estrangeira de *shortest path with negative cycles* – SPNC) e o problema da trilha mínima com restrição de agrupamento (referenciado na literatura estrangeira como *constrained shortest path tour problem* – CSPTP). Para o SPNC, propomos três abordagens exatas baseadas em programação matemática: um modelo compacto de programação linear inteira mista, um algoritmo de *branch-and-bound* especializado e um método de planos de corte. Realizamos uma experimentação englobando tanto instâncias geradas aleatoriamente como também instâncias concebidas por outros autores. Os testes computacionais mostram que as abordagens propostas se sobressaem em relação às técnicas de programação matemática do estado da arte. Além disso, fazemos uma discussão sobre a relaxação linear dos modelos matemáticos presentes na literatura do problema. Com relação ao CSPTP, apresentamos dois modelos compactos para o problema: um de programação linear inteira pura, que chamamos de modelo baseado em vértices artificiais; e outro de programação linear inteira mista, que chamamos de modelo baseado em vértices fronteiras. Para este último, mostramos desigualdades válidas e propomos heurísticas Lagrangeanas determinísticas e não-determinísticas. Experimentos realizados em instâncias da literatura e em outras geradas aleatoriamente validam e atestam a eficácia das nossas contribuições, que alcançam a solução ótima em uma larga quantidade de casos. Mostramos que os modelos baseados em vértices artificais e fronteiras alternam bons resultados dependendo das características de cada instância. A eficiência das metodologias exatas propostas quando comparadas aos algoritmos de *branch-and-bound* especializados, presentes na literatura para o CSPTP, também é comprovada por meio dos testes computacionais, assim como as potencialidades das heurísticas Lagrangeanas, que alcançam a solução ótima para grande parte das instâncias abordadas.

Palavras-chave: Otimização combinatória. Problema do caminho mínimo na presença de ciclos negativos. Problema da trilha mínima com restrição de agrupamento. Programação linear inteira. Relaxação Lagrangeana. Heurísticas.

ABSTRACT

In this work, we study two NP-Hard routing problems: the shortest path with negative cycles (SPNC) and the constrained shortest path tour problem (CSPTP). For the SPNC, we propose three exact approaches based on mathematical programming: a compact mixed integer linear programming model, a specialized branch-and-bound algorithm, and a cutting-plane method. We perform numerical experiments comprising both randomly generated and benchmark instances from the literature. The computational tests show that the proposed approaches stand out from state-of-the-art mathematical programming techniques. Moreover, we discuss the linear relaxations of models present it the literature. Concerning the CSPTP, we show two compact models for the problem: a pure integer linear programming model, which we call dummy node-based model; and a mixed integer linear programming one, which we call frontier node-based model. For the latter, we show valid inequalities and propose deterministic and non-deterministic Lagrangian heuristics. Experiments performed on both randomly generated and benchmark instances from the literature validate and attest the effectiveness of our contributions, which achieve the optimal solution in the vast majority of cases. We show that the dummy node and the frontier node-based models alternate better results depending on the characteristics of each instance. The efficiency over specialized branch-and-bound algorithms from the literature is also proven through experiments, as well as the potentialities behind the Lagrangian heuristics, which find the optimal solution for a large number of instances.

Keywords: Combinatorial optimization. Shortest path with negative cycles. Constrained shortest path tour problem. Integer linear programming. Lagrangian relaxation. Heuristics.

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF SYMBOLS

$\mathbb{R}$	Set of real numbers
G	Undirected graph
D	Directed graph (digraph)
V	Set of vertices (nodes)
E	Set of edges
A	Set of arcs
T	Subset of nodes
c_{ij}	Cost of arc $(i, j) \in A$
s	Source node
t	Destination node
δ_j^+	Set of arcs leaving vertex $j \in V$
δ_j^-	Set of arcs entering vertex $j \in V$

SUMMARY

1 INTRODUCTION

The proposed contributions of this thesis are centered on mathematical programming approaches for NP-Hard constrained shortest path problems, namely, the shortest path problem in digraphs with negative cycle and the constrained shortest path tour problem.

1.1 Context

The shortest path problem (SPP) is one of the most studied problems in network optimization. The SPP is defined on a graph $G = (V,E)$ where V is the set of vertices (or nodes) and E is the set of edges, being each edge assigned to a cost. When one intends to consider only admissible orientations on the edges, set E is replaced by a set of arcs A and graph G turns out to be a directed graph (or digraph) $D = (V,A)$. For the sake of simplicity, we assume the latter case.

Before continuing, we make use of some definitions from graph theory (BONDY *et al.*, 2008) that will guide the reader to better understand fundamental aspects of this work.

Definition 1 *A walk in D is a finite non-null sequence $W = v_0a_1v_1a_2v_2\ldots a_kv_k$, whose terms are alternately nodes and arcs, such that, for $1 \leq i \leq k$, arc $a_i = (v_{i-1}, v_i)$, with $v_{i-1} \in V$ and $v_i \in V$. We say that W is a walk from node v_0 to node v_k, or simply a (v_0, v_k)-walk.*

Definition 2 *A trail is a walk composed of distinct arcs. A trail from $v_0 \in V$ to $v_k \in V$ is referred to as (v_0, v_k)-trail.*

Definition 3 *A path is a walk composed of distinct nodes. A path from $v_0 \in V$ to $v_k \in V$ is referred to as (v_0, v_k)-path.*

The SPP consists in finding a path between two distinct nodes such that the sum of the weights of its arcs is minimized. The problem is often referred to as the single-pair shortest path problem in order to distinguish it from the following variants:

- the single-source shortest path problem, whose aim is to find shortest paths from a source node to all other ones;
- the single-destination shortest path problem, whose aim is to find shortest paths from all nodes to a single destination one;
- the all-pairs shortest path problem, whose aim is to find shortest paths between every pair of nodes.

Several polynomial time algorithms have been devoted to solve the SPP to optimality. Dijkstra's original algorithm (DIJKSTRA, 1959), for instance, is the most popular and it is able to solve the SPP in a digraph with non-negative arc costs. The standard implementation runs in $O(|V|^2)$. An alternative implementation that adopts a Fibonacci heap, conceived by Fredman e Tarjan (1987), is asymptotically the fastest SPP algorithm for that kind of digraph, running in $O(|A|+|V|log|V|)$. Unlike Dijkstra's algorithm, the Bellman–Ford algorithm (FORD, 1956) arises as a versatile routine when computing the SPP in digraphs containing negative arc costs, but without negative (directed) cycles. It runs in $O(|V||A|)$. Still considering the latter kind of digraphs, there are also polynomial time algorithms that compute the SPP between all pairs of nodes, e.g., the Floyd–Warshall algorithm (FLOYD, 1962), running in $O(|V|^3)$, and the Johnson's algorithm (JOHNSON, 1977), running in $O(|V|^2 log|V|+|V||A|)$.

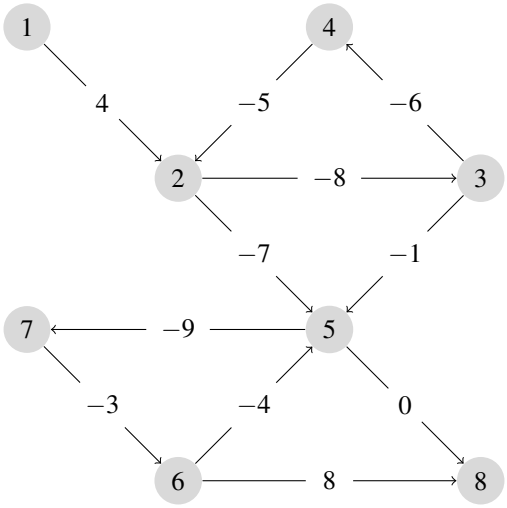
Extensions of the standard SPP have been widely investigated by the scientific community. They commonly add ingredients to the classical SPP resulting in an NP-Hard (GAREY; JOHNSON, 2002) problem, being properly classified as constrained shortest path problems (CSPPs). The most classical CSPP is the resource constrained shortest path problem (RCSPP) (JOKSCH, 1966; HANDLER; ZANG, 1980; BEASLEY; CHRISTOFIDES, 1989), which aims at finding the shortest path between two distinct nodes on a transport network where transversing each arc consumes certain resource whose sum must lie within a given upper limit.

In this work, we study two challenging CSPPs. The first one is the shortest path problem in digraphs with negative cycles (SPNC) (IBRAHIM, 2007), which basically consists in finding the shortest path between two distinct nodes in a digraph containing negative cycles. The second one is the constrained shortest path tour problem (CSPTP) (FERONE *et al.*, 2016; FERONE *et al.*, 2019), which aims at finding the shortest trail between a source and a destination while satisfying visiting constraints on some clusters of nodes.

1.2 Motivation

The first problem addressed here is the SPNC, which is an NP-Hard (BONDY *et al.*, 2008) problem that has been focus of investigation by the literature with the first mathematical programming approach due to Ibrahim (2007). The problem has as input a digraph $D = (V,A)$ with set of nodes V and set of arcs A, with $c_{ij} \in \mathbb{R}$ representing the cost of each arc $(i, j) \in A$. The aim is to determine, if it exists, a path of minimum cost between two distinguished nodes $s \in V$ and $t \in V$. Figures 1 and 2 show an instance and its corresponding optimal solution, respectively.

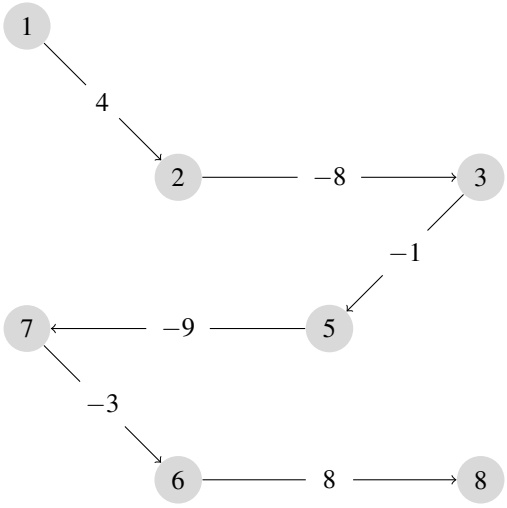
Figure 1 – Example of an SPNC instance, with $s = 1$ and $t = 8$.



Source: The author.

The main motivation we have seen behind the SPNC stems from the fact that there are only few mathematical programming techniques to solve the problem (IBRAHIM *et al.*, 2009; HAOUARI *et al.*, 2013). In this sense, we propose three exact solution approaches. The first one is a compact primal-dual model whose preliminary results were presented at the XLVII Simpósio Brasileiro de Pesquisa Operacional (SBPO 2015) (ANDRADE *et al.*, 2015). We also conceive a combinatorial branch-and-bound algorithm and a cutting-plane method, both presented at the 18th Latin-Iberoamerican Conference on Operations Research (CLAIO 2016) (ANDRADE; SARAIVA, 2016). Extensive computational experiments performed on both benchmark and randomly generated instances indicate that our approaches either outperform or are competitive with existing mixed-integer programming models while providing optimal solution for challenging instances in small execution times. A full version of this work has been

Figure 2 – Optimal solution for the example in Figure 1.



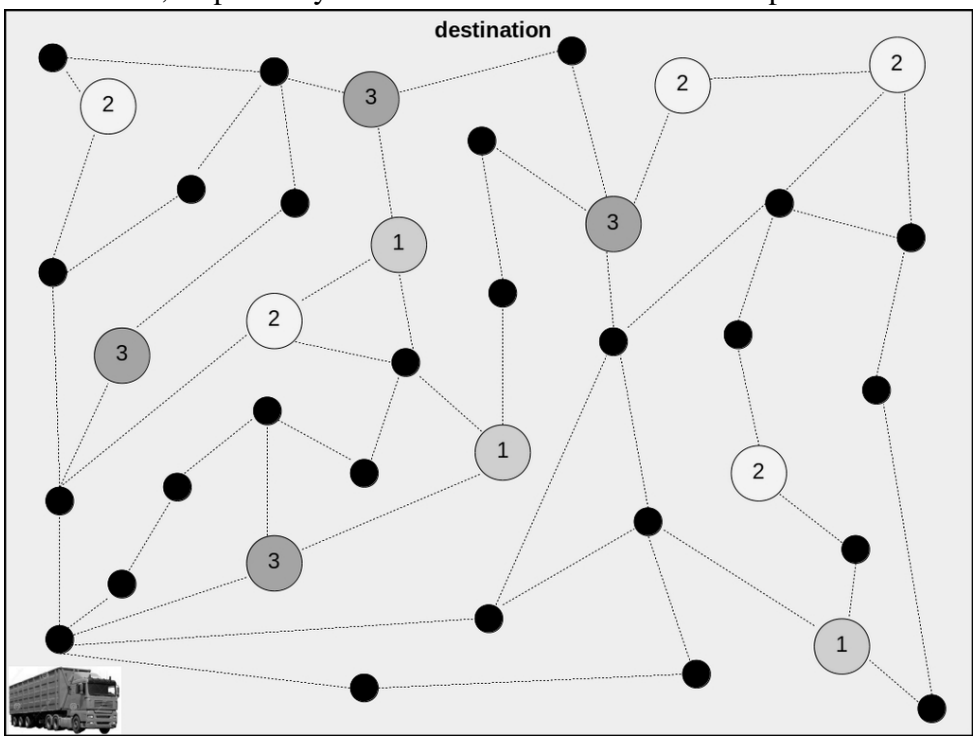
Source: The author.

accepted for publication in Annals of Operations Research (ANDRADE; SARAIVA, 2017).

The second problem under analysis is the CSPTP, an NP-Hard problem that has been recently introduced in the literature (FERONE *et al.*, 2016; FERONE *et al.*, 2019). Defined on a digraph $D = (V,A)$ with set of nodes V and set of non-negative weighted arcs A, the problem is to find the shortest trail between two distinct nodes $s \in V$ and $t \in V$ while visiting at least one element of node disjoint subsets $T_i \subset V$, $i = 1,\ldots,N$, in this order. More precisely, the trail starts at $s \in T_1$ and goes to some element of T_2 (possibly through some intermediate nodes that are not in T_2), then goes to some element of T_3 (possibly through some intermediate nodes that are not in T_3), and so on, until finally ending at $t \in T_N$ (possibly through some intermediate nodes).

Practical applications arise in customers cargo delivery. Each customer has a set of available warehouses to where goods can be delivered (see Figure 3). Goods are stored into a container following a priority queue (see Figure 4) in which customers will be visited.

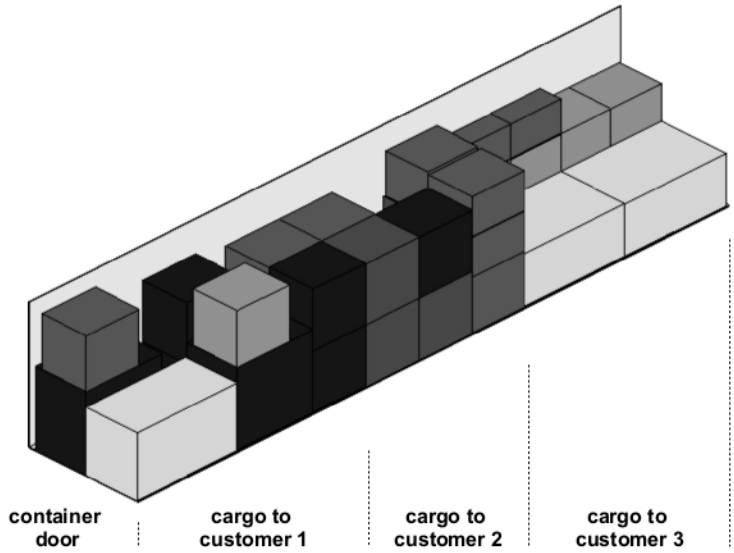
Figure 3 – Transport network: the origin is where the truck starts its delivery service. The cargo of the first customer must be first delivered in any of its warehouse labeled by '1', followed by the second and the third customers in any of their warehouses labeled by '2' and '3', respectively. The route ends at the destination point.



Source: The author.

We present mathematical programming models for the CSPTP. The first one, which we call dummy node-based model, is due to the pilot research presented at the Joint EURO/ALIO International Conference 2018 on Applied Combinatorial Optimization (EURO/ALIO 2018) (AN-

Figure 4 – Loaded container: boxes are delivered according to a precedence order.



Source: The author.

DRADE; SARAIVA, 2018). The second one employs the concept of frontier node and it is further enhanced with valid inequalities. We also develop deterministic and non-deterministic heuristics within a Lagrangian relaxation framework. Experiments performed on benchmark and randomly generated instances compare our results with those obtained by state-of-the-art exact algorithms. A complete version of this study has been submitted to the International Transactions in Operational Research (SARAIVA; ANDRADE, 2019).

1.3 Organization

The remainder of this document is organized as follows. In Chapter 2, we present mathematical programming approaches for the SPNC, which comprise a compact formulation, a combinatorial branch-and-bound algorithm, and a cutting-plane method. In Chapter 3, we address the CSPTP by proposing compact formulations, valid inequalities and a Lagrangian-based heuristic framework. In Chapter 4, we close the text drawing conclusions concerning our work and giving directions for future research.

2 SHORTEST PATH WITH NEGATIVE CYCLES

We devote this chapter to the study of the shortest path with negative cycles. Problem preliminary studies were presented at SBPO 2015 (ANDRADE *et al.*, 2015) and CLAIO 2016 (ANDRADE; SARAIVA, 2016). The complete work was accepted in the Annals Operations Research (ANDRADE; SARAIVA, 2017).

2.1 Introduction

Let $D = (V,A)$ be a digraph with a set V of vertices, and a set A of arcs. An arc from vertex $i \in V$ to vertex $j \in V$ is represented as $(i, j) \in A$, with $c_{ij} \in \mathbb{R}$ being its cost, for all $(i, j) \in A$. Given two distinct vertices $s \in V$ and $t \in V$, the problem of finding the shortest path with negative cycles (SPNC) consists in determining, if it exists, an (s,t)-path of minimum cost in D, a digraph possibly containing (directed) cycles whose sum of their arc costs is negative. This NP-Hard problem (BONDY *et al.*, 2008) has applications in telecommunications and recently gained renewed interest in the literature (IBRAHIM *et al.*, 2009; DREXL, 2013; HAOUARI *et al.*, 2013; IBRAHIM *et al.*, 2015b; IBRAHIM, 2015; IBRAHIM *et al.*, 2015a; TACCARI, 2016).

We find related problems in digraphs containing cycles with negative cost. For instance, the negative cost cycle detection problem (YAMADA; KINOSHITA, 2002; SUBRAMANI; KOVALCHICK, 2005; SUBRAMANI, 2007; GU *et al.*, 2009; HOUGARDY, 2010); the all-pairs shortest path problem (MEHLHORN *et al.*, 2002); the single-source all-destinations shortest path problem (PUGLIESE; GUERRIERO, 2016), who conceive a dynamic multi-dimensional labelling approach that runs in $O(|V|2^{2|V|})$; and the minimum weighted elementary directed cycle problem (IBRAHIM *et al.*, 2016), to cite just a few problems. Well-established algorithmic techniques as the Bellman-Ford or the Dijkstra's algorithms, as well as dynamic programming, are in charge of solving them. It is known that Bellman-Ford algorithm fails when determining shortest paths in digraphs with negative cycles. Hence, handling, e.g. the SPNC with an algorithm is not trivial, thus requiring mathematical programming techniques.

In this sense, we propose three exact solution approaches for the SPNC. An enhanced MTZ-based compact primal-dual model, a specialized combinatorial branch-and-bound algorithm, and a cutting-plane method. The ideas we explore in these solution approaches are new for the problem and constitute our main contributions. The algorithms are easy-to-implement and allowed to realize an extensive set of computational experiments conducted on both benchmark

and new randomly generated instances. The proposed approaches either outperform or are competitive with existing solution strategies for the considered instances. Moreover, this chapter reports a discussion comparing the linear relaxation of the MTZ-primal-dual model with those of models from the literature, as well as presents a new result concerning the quality of the integrality gap for particular instances of the problem when solved by the MTZ-based model.

The remainder of this chapter is organized as follows. Section 2.2 reports mixed-integer programming (MIP) models for the SPNC. Section 2.3 presents an enhanced MTZ-primal-dual formulation and gives some properties concerning its linear relaxation. Sections 2.4 and 2.5 describe a combinatorial branch-and-bound algorithm and a cutting-plane strategy for the problem, respectively. Section 2.6 reports computational experiments on benchmark and new randomly generated instances. Finally, Section 2.7 gives the conclusions of this chapter.

2.2 SPNC formulations

For the sake of convenience, we adopt the following convention throughout the text. Given $D = (V,A)$, let δ_j^+ (resp., δ_j^-) denote the set of arcs leaving (resp., entering) vertex $j \in V$.

2.2.1 The linear flow-based model of Ibrahim et al. (2009)

Ibrahim *et al.* (2009) propose a compact MIP model based on a non-simultaneous network flow stated as follows. Let x_{ij} (resp., y_j) be a binary variable taking value 1 if arc $(i,j) \in A$ (resp., vertex $j \in V$) belongs to the solution, or 0 otherwise; and let $z_{ij}^k \geq 0$ be a continuous variable representing the flow through arc $(i,j) \in A$, in an (s,t)-path, from vertex s to vertex $k \in V \setminus \{s\}$. The model (IMM) is the following.

$$\text{(IMM)} \quad \min \quad \sum_{(i,j)\in A} c_{ij} x_{ij} \tag{2.1}$$

$$\text{s.t.} \quad \sum_{j\in\delta_s^+} x_{sj} = 1 \tag{2.2}$$

$$\sum_{j\in\delta_t^-} x_{jt} = 1 \tag{2.3}$$

$$\sum_{j\in\delta_i^+} x_{ij} = y_i, \quad \forall i \in V\setminus\{s,t\} \tag{2.4}$$

$$\sum_{j\in\delta_i^-} x_{ji} = y_i, \quad \forall i \in V\setminus\{s,t\} \tag{2.5}$$

$$\sum_{j\in\delta_s^+} z_{sj}^k - \sum_{j\in\delta_s^-} z_{js}^k = y_k, \quad \forall k \in V\backslash\{s\} \tag{2.6}$$

$$\sum_{j\in\delta_i^+} z_{ij}^k - \sum_{j\in\delta_i^-} z_{ji}^k = 0, \quad \forall k \in V\backslash\{s\},\ i \in V\backslash\{s,k\} \tag{2.7}$$

$$\sum_{j\in\delta_k^+} z_{kj}^k - \sum_{j\in\delta_k^-} z_{jk}^k = -y_k, \quad \forall k \in V\backslash\{s\} \tag{2.8}$$

$$z_{ij}^k \leq x_{ij}, \quad \forall k \in V\backslash\{s\},\ (i,j) \in A \tag{2.9}$$

$$x_{ij} \in \{0,1\}, \quad \forall (i,j) \in A \tag{2.10}$$

$$y_j \in \{0,1\}, \quad \forall j \in V \tag{2.11}$$

$$z_{ij}^k \geq 0, \quad \forall k \in V\backslash\{s\},\ (i,j) \in A \tag{2.12}$$

Constraints (2.2) and (2.3) establish that one arc leaves the source s and one arc enters the destination t, respectively. Constraints (2.4) and (2.5) state that the number of arcs leaving and entering vertex $i \in V\backslash\{s,t\}$, respectively, is equal to 1 if it belongs to the (s,t)-path; otherwise, no arc is incident to the vertex. The flow conservation constraints (2.6) and (2.8) state, respectively, that if vertex k is in the solution, then the amount of flow leaving s to k is 1 and the amount of flow arriving at k is also 1; otherwise, these flows are null. Constraints (2.7) state that transit vertices $i \in V\backslash\{s,k\}$ do not retain flow from s to k. Constraints (2.9) state that if an arc $(i,j) \in A$ is not in the solution, then the flow in this arc is null. The remaining constraints (2.10)–(2.12) indicate the domain of the variables. Model (IMM) contains $O(|V|^2)$ binary variables, $O(|V|^3)$ continuous variables and $O(|V|^3)$ constraints.

Ibrahim *et al.* (2015b) and Ibrahim (2015) adopt model (IMM) to investigate some classes of valid inequalities and lifting techniques within a cutting-plane framework, whose performances are analyzed for grid instances. The reason why they consider the above flow-based model stems from the fact that it provides very strong initial linear programming (LP) relaxations for the problem.

2.2.2 The non-linear model of Haouari et al. (2013)

This section reproduces the compact non-linear MIP model of Haouari *et al.* (2013). Consider the characteristic vector x of arc decision variables as in model (IMM). Let non-negative continuous variables u_j, $j \in V$, indicate the number of arcs from the source s to a vertex $j \in V\backslash\{s\}$ in the (s,t)-path, with $u_s = 0$.

Assumption 1 $|\delta_s^-| = |\delta_t^+| = 0$, *and* $(s,t) \notin A$.

Considering Assumption 1, Haouari *et al.* (2013) propose the following model.

$$\text{(HMM)} \quad \min \quad \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{2.13}$$

$$\text{s.t.} \quad (2.2),(2.3),(2.10), \textit{ and}$$

$$\sum_{i\in\delta_j^-} x_{ij} - \sum_{i\in\delta_j^+} x_{ji} = 0, \quad \forall j \in V\backslash\{s,t\} \tag{2.14}$$

$$\sum_{i\in\delta_j^-} x_{ij} \leq 1, \quad \forall j \in V\backslash\{s,t\} \tag{2.15}$$

$$u_j x_{ij} = (u_i+1)x_{ij}, \quad \forall j \in V\backslash\{s,t\},\ i \in \delta_j^-\backslash\{s\} \tag{2.16}$$

$$u_j x_{sj} = x_{sj}, \quad \forall j \in \delta_s^+ \tag{2.17}$$

$$u_s = 0, \quad 1 \leq u_j \leq |V|-1, \quad \forall j \in V\backslash\{s\} \tag{2.18}$$

Constraints (2.14) establish that the number of arcs entering any vertex $j \in V\backslash\{s,t\}$ is equal to the number of arcs leaving this vertex. Constraints (2.15) impose that at most one arc enters each vertex $j \in V\backslash\{s,t\}$. Constraints (2.16) and (2.17) avoid cycles in any feasible solution. The idea in (2.16) is to impose that if arc (i,j) is in the solution, then the distance from s to j, say u_j, is one unit more than the distance from s to i, say u_i. Constraints (2.17) impose that the first vertex to be visited will be at a distance 1 from the source s. Finally, constraints (2.18) define the domain of the u variables.

To linearize constraints (2.16) and (2.17), the authors use the reformulation linearization technique (RLT) of Sherali e Adams (1990), which is accomplished by substitutions of variables $\alpha_{ij} = u_j x_{ij}$, and $\beta_{ij} = u_i x_{ij}$, for all $j \in V\backslash\{s\}$, and $i \in \delta_j^-\backslash\{s\}$ (the reader is referred to Haouari *et al.* (2013) for further details). Thus, they obtain the following model.

$$\text{(HMM-RLT)} \quad \min \quad \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{2.19}$$

$$\text{s.t.} \quad (2.2),(2.3),(2.10),(2.14),(2.15), \textit{ and}$$

$$\alpha_{ij} = \beta_{ij} + x_{ij}, \quad \forall j \in V\backslash\{s\},\ i \in \delta_j^-\backslash\{s\} \tag{2.20}$$

$$x_{sj} + \sum_{i\in\delta_j^-\backslash\{s\}} \alpha_{ij} - \sum_{i\in\delta_j^+} \beta_{ji} = 0, \quad \forall j \in \delta_s^+ \tag{2.21}$$

$$\sum_{i\in\delta_j^-} \alpha_{ij} - \sum_{i\in\delta_j^+} \beta_{ji} = 0, \quad \forall j \in V\backslash\{\delta_s^+ \cup \{s,t\}\} \tag{2.22}$$

$$x_{ij} \leq \alpha_{ij} \leq (|V|-1)x_{ij}, \quad \forall j \in V \backslash \{s\},\ i \in \delta_j^- \backslash \{s\} \tag{2.23}$$

$$x_{ij} \leq \beta_{ij} \leq (|V|-1)x_{ij}, \quad \forall j \in V \backslash \{s\},\ i \in \delta_j^- \backslash \{s\} \tag{2.24}$$

Constraints (2.20)–(2.24) work as subtour elimination constraints. Model (HMM-RLT) contains $O(|V|^2)$ binary variables, $O(|V|)$ continuous variables and $O(|V|^2)$ constraints.

Models (IMM) and (HMM-RLT) are the core of the main solution approaches for the SPNC. They are used for comparison purpose with the solution approaches in the sequel.

2.3 Enhanced MTZ-primal-dual model

Miller-Tucker-Zemlin (MTZ) subtour elimination constraints (MILLER *et al.*, 1960) have been recently explored in MIP models to solve the SPNC. For instance, Andrade *et al.* (2015) and Taccari (2016) conceive a compact primal-dual model for this problem based on the classical MTZ constraints. We explore ideas proposed by Desrochers e Laporte (1991) to obtain a strengthened model for the problem. Initially, consider the following property.

Property 1 *Let $v_1, v_2, v_3, \ldots, v_{n-1}, v_n$ be a sequence of vertices in an (s,t)-path, with $v_1 = s$ and $v_n = t$; and u_v be the number of arcs from the source s to v in this path, for all $v \in \{v_1, v_2, v_3, \ldots, v_{n-1}, v_n\}$. Thus, $u_{v_1} = 0$ and $u_{v_k} = u_{v_{k-1}} + 1$, for $k = 2, 3, \ldots, n$.*

Our first solution approach for the SPNC uses Property 1 with the aim of introducing a new SPNC formulation that connects primal variables x and dual variables u in a unique model, focusing on those variables directly involved in the (s,t)-path. The model, referred to as (CPD), is the following:

$$\text{(CPD)} \quad \min \quad \sum_{(i,j) \in A} c_{ij} x_{ij} \tag{2.25}$$

$$\text{s.t.} \quad (2.2), (2.3), (2.10), (2.14), (2.15), (2.18), \textit{ and}$$

$$u_i - u_j + M x_{ij} \leq M - 1, \quad \forall (i,j) \in A,\ i \neq j \tag{2.26}$$

Constraints (2.26) are MTZ subtour elimination constraints, with M being a big-M positive constant (e.g., $M = |V|$ for SPNC). They avoid cycles in any feasible solution, ensuring that if an arc $(i,j) \in A$ belongs to the solution, then $u_j - u_i \geq 1$. We can show, for vertices from

s to t in any feasible solution, that Property 1 is verified. The compact model (CPD) contains $O(|V|^2)$ binary variables, $O(|V|)$ continuous variables and $O(|V|^2)$ constraints.

Proposition 1 *Model (CPD) correctly returns an (s,t)-path of minimum cost.*

Proof 1 *Straightforward.*

Following ideas from Desrochers e Laporte (1991), we can strengthen model (CPD) while replacing constraints (2.26), when both arcs $(i,j),(j,i)$ belong to A, by:

$$u_i - u_j + (|V|-1)x_{ij} + (|V|-3)x_{ji} \leq |V|-2, \; \forall \, (i,j),(j,i) \in A,$$

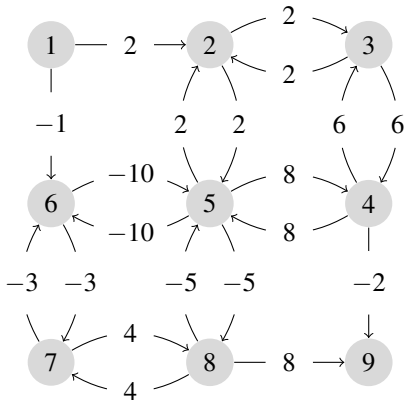
In what follows, we refer to model (CPD) as the proposed model enhanced with the aforementioned constraints.

2.3.1 Remarks on linear relaxed solutions of SPNC models

We investigate the quality of the linear relaxed solutions for the three SPNC models (IMM), (HMM-RLT), and (CPD). Basically, we give some examples where the value of the linear relaxed solution of model (HMM-RLT) is better than that one of (CPD), and vice-versa. We also prove that model (IMM) is tighter than (CPD).

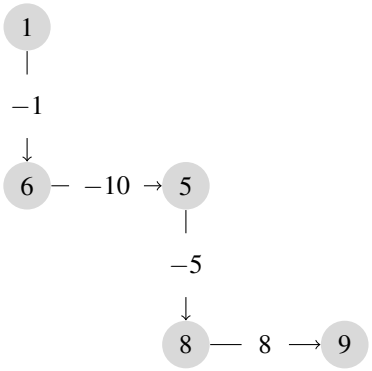
In Figure 6, the shortest $(1,9)$-path has cost equal to -8 for the digraph in Figure 5. The linear relaxed solution for (CPD) is depicted in Figure 7, and has a value equal to -10 while that one for (HMM-RLT), in Figure 8, has a value equal to -10.66. The value $u_s = 0$ is omitted, as well as variable values for α and β that are not defined for arcs leaving s. For this example, the linear relaxed solution value obtained by (CPD) is larger than that one obtained by (HMM-RLT).

Figure 5 – Instance: a digraph with $s = 1$ and $t = 9$. Values in the middle of the arcs represent their costs.



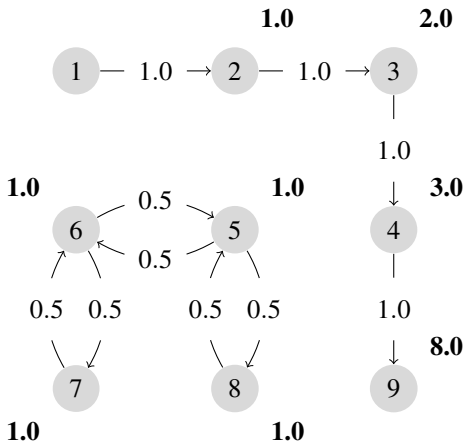
Source: The author.

Figure 6 – Shortest $(1,9)$-path of cost equal to -8.



Source: The author.

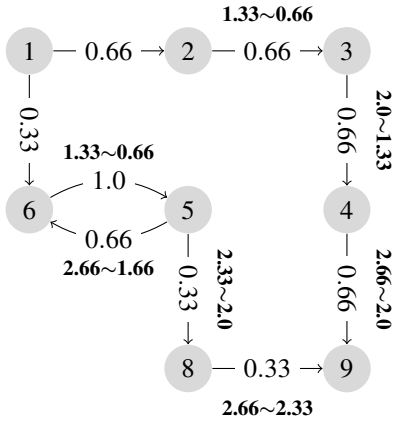
Figure 7 – Optimal linear relaxed solution for model (CPD) of cost equal to -10. Values in the middle of the arcs (resp., in bold font near each vertex) denote the corresponding values of the variables x (resp., u).



Source: The author.

On the other hand, in Figure 10, the shortest $(1,9)$-path for the digraph in Figure 9 has cost equal to -28. The linear relaxed solution for models (CPD) and (HMM-RLT) are

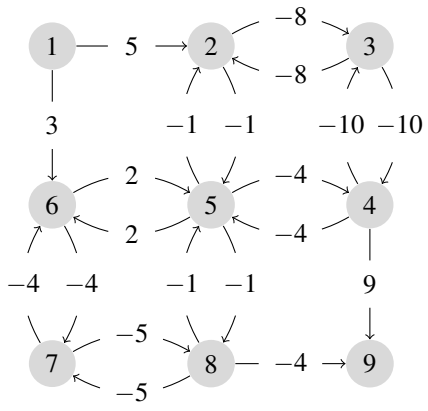
Figure 8 – Optimal linear relaxed solution for model (HMM-RLT) of cost equal to -10.66. Values in the middle of the arcs (resp., in bold font near each arc) denote the corresponding values of variables x (resp., α and β, represented as $\alpha \sim \beta$).



Source: The author.

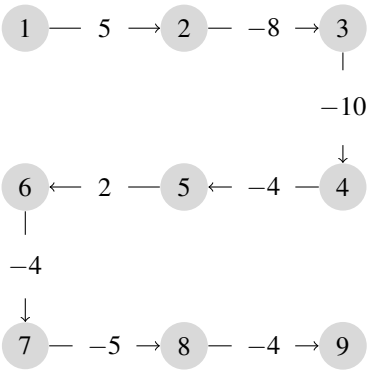
depicted in Figures 11 and 12, and their values are equal to -33 and -31.71, respectively. The arc attribute notation is the same as in Figures 5–8. Here, the linear relaxed solution value obtained by (HMM-RLT) is larger than that one obtained by (CPD).

Figure 9 – Instance: a digraph instance with $s = 1$ and $t = 9$.



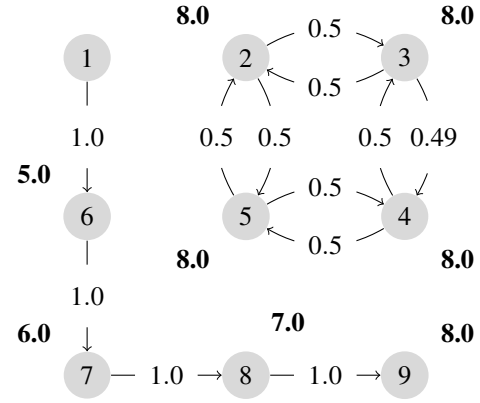
Source: The author.

Figure 10 – Shortest $(1,9)$-path of cost equal to -28.



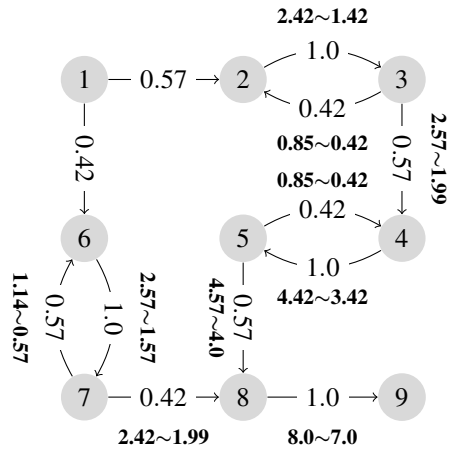
Source: The author.

Figure 11 – Optimal linear relaxed solution for model (CPD) of cost equal to -33.



Source: The author.

Figure 12 – Optimal linear relaxed solution for model (HMM-RLT) of cost equal to -31.71.



Source: The author.

Proposition 2 *When projecting out y and z variables from model (IMM), the set of feasible linear relaxed solutions of the resultant model is contained in that one of (CPD) when projecting out its u variables.*

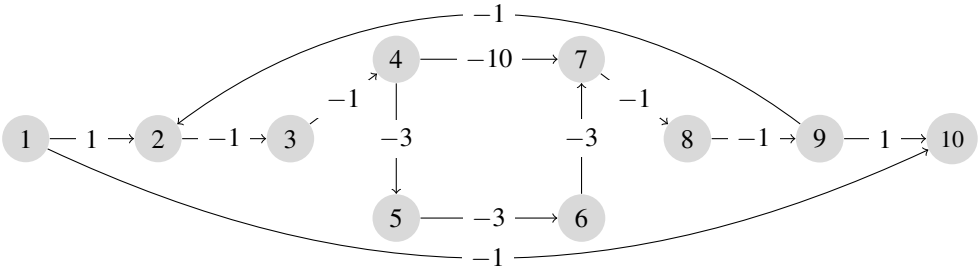
Proof 2 *For every solution $(\bar{x},\bar{z},\bar{y})$ of the linear relaxation of (IMM), the digraph induced by the non-null components of $\bar{x} \in \mathbb{R}_{+}^{|A|}$ is always connected. This digraph does not contain a cycle induced by the components of $\bar{x}$ equal to* 1*. If it were the case, by* (2.4)–(2.5)*, there should exist exactly one arc in this digraph leaving (or entering) each vertex of such cycle. But, by constraints* (2.6)–(2.8)*, we have at least one vertex not belonging to this cycle that is connected to one of its vertices, thus, violating one of the constraints* (2.4) *or* (2.5)*. In this case, any cycle in a linear relaxed solution of (IMM) is induced by fractional components of $\bar{x}$. We use $\bar{x}$ to construct a feasible solution to the linear relaxation of (CPD) as follows. We set $\bar{u}_s = 0$ and, for all $\bar{x}_{i,j} = 1$, we set $\bar{u}_j = \bar{u}_i + 1$, with the value of $\bar{u}_i$ determined accordingly without violating* (2.18) *and* (2.26)*. In case $0 < \bar{x}_{i,j} < 1$, it is always possible to determine a pair of values for $\bar{u}_i$ and $\bar{u}_j$ in order to satisfy* (2.26) *since these constraints were conceived (MILLER et al., 1960) to become inactive when $\bar{x}_{i,j} \neq 1$. The remaining constraints of (CPD) are clearly satisfied by $\bar{x}$.*

Proposition 3 *When projecting out u variables from model (CPD), the set of feasible linear relaxed solutions of the resultant model is not contained in that one of (IMM) when projecting out its y and z variables.*

Proof 3 *To prove this result, we claim that there are optimal linear relaxed solutions of (CPD) that induce disconnected digraphs, which cannot be feasible to the linear relaxation of (IMM).*

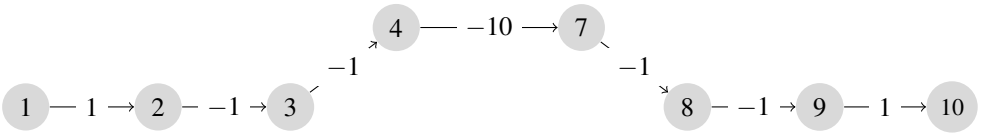
An example of Proposition 3 is given for the digraph in Figure 13. The notation is the same as before. The linear relaxed solution for (CPD) is depicted in Figure 15. It is not connected and has cost equal to -15.5 (with $M = 12$ in (2.26)) while that one for (IMM) is the shortest $(1,10)$-path of cost equal to -12 in Figure 14.

Figure 13 – Instance: a digraph with $s = 1$ and $t = 10$.



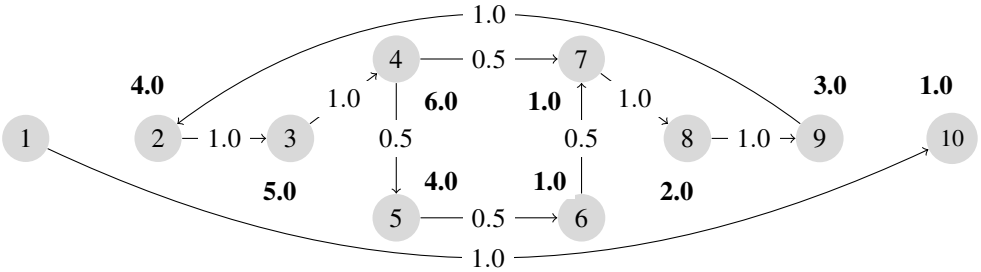
Source: The author.

Figure 14 – Shortest $(1,10)$-path of cost equal to -12. It is feasible for the linear relaxation of model (IMM).



Source: The author.

Figure 15 – Optimal linear relaxed solution for model (CPD) of cost equal to -15.5.



Source: The author.

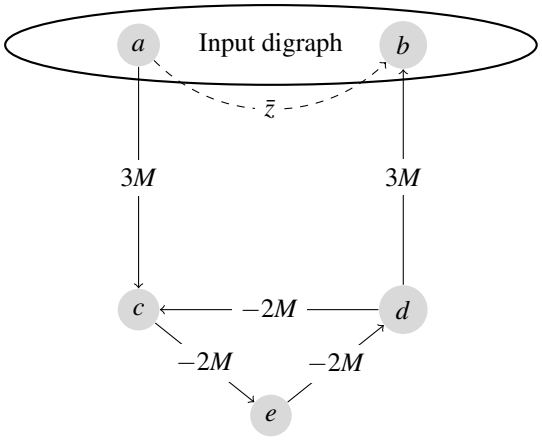
Proposition 4 *We can construct SPNC instances whose integrality gap for model (CPD) is as large as possible.*

Proof 4 *We prove this result by constructing an instance presenting a very large integrality gap for model (CPD). For instance, consider any SPNC instance. Let us modify this instance by adding an extra structure as done in Figure 16 to its input digraph delimited by the ellipse. Assume that the optimal (a,b)-path cost in the input digraph (represented by the dashed (a,b) arc) and its optimal linear relaxed solution value are denoted by $\bar{z}$ and $\bar{z}_R$, respectively. The linear relaxed solution for model (CPD) of the modified digraph presents arc decision variables associated with the structure equal to e.g. $x_{ac} = x_{db} = 0$ and $x_{ce} = x_{ed} = x_{dc} = 0.83$, whose values are for $M = |V|$. Labels related to u variables are equal to $u_c = u_d = u_e = 0$. The cost referred to the added structure is $-6M \times 0.83$. Thus, the new linear relaxed solution cost is limited from above by $\bar{z} - 6M \times 0.83$. Therefore, the integrality gap of the modified digraph increases with the value of M while its optimal solution remains the same.*

2.4 Combinatorial branch-and-bound algorithm

A combinatorial branch-and-bound (B&B) algorithm works as usual with its classical branching, bounding, and pruning techniques, except for the way we proceed to branch on integer B&B node solutions from the combinatorial relaxation employed to evaluate nodes in the B&B search tree.

Figure 16 – Adapting any SPNC instance to have a very large integrality gap for model (CPD). M is a big positive value. Vertices a and b are the given path source and destination vertices, respectively, in the input digraph. The structure associated with vertices $\{c,d,e\}$ modifies the input digraph.



Source: The author.

In the next paragraphs, we detail the main bounding and branching operations of our specialized B&B algorithm for the SPNC.

2.4.1 Evaluation of B&B nodes

In order to obtain a relaxation of the SPNC to be used in a B&B algorithm, we drop the subtour elimination constraints (2.26) from model (CPD), as well as the integrality requirement on the x variables. Thus, the following model evaluates each node of the B&B search tree.

$$\text{(CPD-CR)} \quad \min \quad \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{2.27}$$

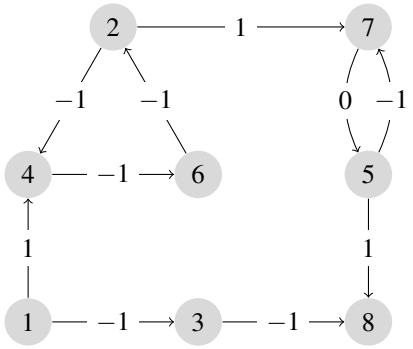
$$\text{s.t.} \quad (2.2),(2.3),(2.14),(2.15), \textit{ and}$$

$$0 \leq x_{ij} \leq 1, \quad \forall (i,j) \in A \tag{2.28}$$

Based on Proposition 5, we see that model (CPD-CR) is a combinatorial optimization problem since any feasible solution for (CPD-CR) is an (s,t)-path with a set of disjoint cycles (which are also disjoint w.r.t. the path from s to t), as we can see e.g. in Figures 17 and 18. If there is no cycle in the optimal solution of (CPD-CR), then it is feasible for (CPD).

Proposition 5 *The constraint matrix of model (CPD-CR) is totally unimodular (TU).*

Figure 17 – Input digraph: $s = 1$ and $t = 8$.



Source: The author.

Proof 5 *Let $A_{(CPD-CR)}$ be the coefficient matrix associated with the constraints of (CPD-CR). Consider $M = \{1,\ldots,2|V|-2\}$ as the set of row indices, $N = \{1,\ldots,|A|\}$ as the set of column indices, and a_{ij} as the element belonging to the i-th row and j-th column of $A_{(CPD-CR)}$. We resort to the characterization of TU matrices conceived by Ghouila-Houri (1962): $\forall I \subseteq M$, $\exists\ I_1, I_2 \in I$ such that $\sum_{i\in I_1} a_{ij} - \sum_{i\in I_2} a_{ij} \in \{0,\pm 1\}$, $\forall j \in N$. The proof is by construction. Let $M_1 = \{1,\ldots,|V|\}$ and $M_2 = \{|V|+1,\ldots,2|V|-2\}$ be partitions of M, and let I be an arbitrary subset of M. We construct I_1 and I_2 as follows. $I_1 = \{I \cap M_1\} \cup \{i \in M_2 : (i-|V|+1) \notin I\}$, $I_2 = \{i \in M_2 : (i-|V|+1) \in I\}$. The rationale behind the partition scheme is that rows from flow conservation constraints are all added to I_1, while rows from degree constraints are added to either I_1 or I_2 depending on which rows from flow conservation constraints belong to subset I. Take an arbitrary vertex $i \in V\backslash\{s,t\}$. The row defining the degree constraint of vertex i, if in I, is added to I_1 if and only if the row defining the flow conservation constraint of i is not in I. If the latter row is in I and consequently in I_1, then adding the former row to I_1 would produce a sum of value equal to ± 2.*

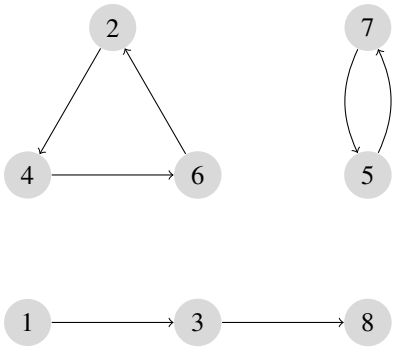
An example of a coefficient matrix, with respect to (CPD-CR), for a complete digraph with $|V| = 4$ is detailed in Annexe A.

A straightforward consequence of Proposition 5 is that the solution of model (CPD-CR), with respect to a given node in a B&B tree, is integer. Thus, if the solution is cycle-free, it provides an upper bound on the optimal solution value of the original problem.

2.4.2 Combinatorial branch

In the B&B algorithm, when a relaxed node solution contains disjoint cycles (see e.g. Figures 17 and 18), we represent all of them by $C_1, C_2, \ldots, C_m$. Note that the structure of the digraph obtained from the solution of (CPD-CR) allows us to recover cycles by using a simple

Figure 18 – Optimal solution when solving the example of Figure 17 with (CPD-CR): an (s,t)-path and two disjoint cycles.

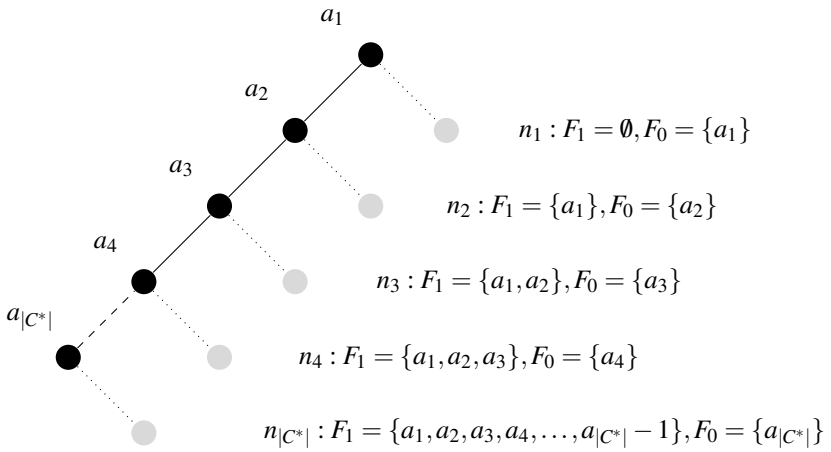


Source: The author.

depth-first search (DFS) polynomial-time routine. We then proceed by choosing one of them to be broken. In particular, our algorithm selects the cycle $C^* \leftarrow C_j$ $(1 \leq j \leq m)$ with the minimum number of arcs (in Figure 18, the smallest cycle is the one represented by arcs $(5,7)$ and $(7,5)$). This is intended to reduce the number of new partitions during the branching process of the B&B algorithm, as explained below (priority is given to the cycle with the largest cost in case of ties). Once determined the cycle C^*, let $a_1, a_2, a_3, \ldots, a_{|C^*|}$ be its arcs. The corresponding B&B node is partitioned into $|C^*|$ new subproblems according to the following idea. In the first partition, we fix arc a_1 out of the solution; in the i-th partition, for $i \in \{2, \ldots, |C^*|\}$, we fix arc a_i out of the solution and impose that all arcs $a_1, a_2, a_3, \ldots, a_{i-1}$ belong to its solution. This is called a disjunctive combinatorial branch and has been explored within B&B schemes in case of B&B node relaxed solutions are integer (ANDRADE; FREITAS, 2013; BATSYN *et al.*, 2013).

We illustrate this branching process in Figure 19. Solid (resp., dotted) lines show that an arc is fixed in (resp., fixed out of) the corresponding B&B node solution. All new partitions

Figure 19 – Disjunctive combinatorial branch. New partitions $n_1, n_2, \ldots, n_{|C^*|}$ are the leaves on this tree.



Source: Artwork adapted from Andrade e Freitas (2013).

(subproblems) are represented by the leaves in the corresponding binary tree representation. Fixing an arc in (resp., out of) a B&B node relaxed solution can be done by setting each corresponding arc decision variable lower and upper bounds to one (resp., to zero). In this figure, for every new subproblem, F_0 and F_1 are the sets of arcs whose both lower and upper bounds of their corresponding arc decision variables are fixed at zero and at one, respectively. Every new subproblem is evaluated before its inclusion into the B&B tree of open subproblems. We use the best-first criterion to choose the next open node to be evaluated, which explores the B&B nodes in an increasing order of relaxed solution values.

In the next section, we devise a cutting-plane algorithm to solve (CPD) based on the solution of model (CPD-CR).

2.5 Cutting-plane method

We present an exact cutting-plane (CP) method for the SPNC that is based on model (CPD-CR). This model is a relaxation for (CPD) and its solution can contain cycles. Thus, the idea is to iteratively cut off cycles from the space of feasible solutions of (CPD-CR). Remember that we can detect a cycle in a given digraph by using, for instance, a DFS procedure. In this case, we determine the corresponding digraph induced by any feasible solution $\bar{x}$ of (CPD-CR) and run a DFS-based cycle detection procedure for this solution, say `containCycle(`$\bar{x}$`)`. This procedure returns all cycles present in $\bar{x}$. We generate subtour elimination constraints for every cycle in $\bar{x}$ and add them to model (CPD-CR), then re-optimized.

This process is repeated until the solution of (CPD-CR) presents no cycle, when the optimal solution of (CPD) is achieved. Observe that, as subtour elimination constraints are

Algorithm 1: Cutting-plane method for the SPNC.

1: **Data: SPNC instance.**
2: **Result: SPNC optimal solution $\bar{x}$.**
3: $\bar{x} \leftarrow$ `solve(CPD-CR);`
4: **while** (`containCycle()`) **do**
5: Let $C_1, C_2, \ldots, C_m$ represent the cycles induced by $\bar{x}$;
6: **for** ($i \leftarrow 1$ **to** m) **do**
7: Let a_j be a decision variable associated with arc j, for all $j \in C_i$;
8: Add inequality $\sum_{j \in C_i} a_j \leq |C_i| - 1$ to (CPD-CR);
9: **end for**
10: $\bar{x} \leftarrow$ `solve(CPD-CR);`
11: **end while**

iteratively included in model (CPD-CR), the total unimodularity of its constraint matrix may be violated. When a fractional solution is obtained, we stop working with continuous variables on (CPD-CR) and start adopting the integrality requirement to guide the CP method.

Algorithm 1 gives the main steps of our CP method, as well as the inequality implemented to cut off cycles. This algorithm uses the procedure `containCycle` defined above.

The number of cycles in a digraph D is finite, although exponential in the worst case. Thus, Algorithm 1 always ends with an optimal solution for model (CPD) after enumerating at most all cycles of D.

2.6 Computational experiments

We implemented models (IMM) and (HMM-RLT) of Ibrahim *et al.* (2009) and Haouari *et al.* (2013), respectively, as well as model (CPD), the combinatorial B&B, and the cutting-plane (CP) algorithm. All five solution approaches are compared in this chapter. We report numerical results for problem instances categorized as follows: 38 general digraphs of Haouari *et al.* (2013) (in Table 1), 50 randomly generated digraphs (in Table 2), 17 grid digraphs of Ibrahim *et al.* (2015b) (in Table 3), 90 pricing subproblems of Taccari (2016) (in Tables 4 and 5), and 9 new instances (in Table 6) obtained from instances from Tables 1 and 2. For all instances, we set $|\delta_s^-| = |\delta_t^+| = 0$ and $(s,t) \notin A$ in order to apply properly (HMM-RLT). In (IBRAHIM *et al.*, 2015b), the authors give the number of negative cycles for their instances. As it was very time consuming to determine the same parameter for the remaining instances we deal with, probably due to their exponential number of negative cycles, this information was not considered here. Our experiments were performed on a PC Intel Pentium i7 / 8 x 3.60 GHz / 16 GB DDR3 RAM under Linux Ubuntu 14.05 LTS. We used CPLEX 12.6 and Java concert technology. The CPU time limit for each instance was set to 3600 seconds.

Haouari *et al.* (2013) randomly generate 38 digraphs as follows. First, for each vertex j ($j = 1,\ldots,|V|-1$), the out-degree d_j^+ of vertex j is drawn uniformly at random from $[1, min\{3, |V|-j-1\}]$. Second, d_j^+ outgoing arcs from j ($j = 1,\ldots,|V|-1$) are created by randomly choosing d_j^+ endpoints in $\{j+1,\ldots,|V|\}$. If, after this process, there is a vertex j ($j = 2,\ldots,|V|$) having no incident arc in the resulting partial graph, then an arc (i,j) is arbitrarily created, with i randomly drawn from $\{1,\ldots,j-1\}$. To create cycles, for each vertex j ($j = 2,\ldots,|V|$), an arc (j,k) is added to the digraph with probability equal to $\frac{1}{2}$, with k being randomly chosen from $\{2,\ldots,j-1\}$. Finally, for each arc $(i,j) \in A$, the corresponding cost c_{ij}

is initially drawn uniformly at random from $[1,50]$, and then multiplied by -1 with probability equal to $\frac{2}{3}$ to impose the existence of negative cycles. They always set $s = 1$ and $t = |V|$.

Ibrahim *et al.* (2015b) generate 17 digraphs (grid instances) with a random arc orientation. Arc cost values are uniformly drawn from $[-13,7]$. Vertices s and t are also randomly chosen.

Taccari (2016) reports 90 pricing subproblems from a column-generation algorithm for the asymmetric m-salesmen TSP. The pricing problem in such an algorithm is a shortest path problem on a graph with negative cycles. Arc cost values lie in $[-10^8, 30000]$. These instances contain at least one negative cycle.

We also randomly generate 50 digraphs using the following procedure. For each pair of vertices $i \in V$ $(i \neq |V|)$ and $j \in V$ $(j \neq 1)$, with $i \neq j$, an arc (i, j) is created according to a given input probability ρ, whose value corresponds to the desired arc density of the digraph that can be inferred by $|A|/|V|(|V|-1)$. Arc costs c_{ij} are drawn uniformly at random from $[-48,50]$. Vertices s and t are set to 1, and to $|V|$, respectively. We perform a DFS-based post-processing operation to check the connectivity of our instances. Disconnected digraphs are not considered.

Concerning the instances used in Table 6, they were obtained from the last four instances from Table 1 and from the last five instances from Table 2. These instances with small integrality gaps were modified by adding to them the structure presented in Figure 16 with $M = |V|$. This is intended to make their integrality gaps very large for model (CPD) as stated by Proposition 4, thus allowing us to evaluate if such a change has some impact on the performance of all solution approaches for these instances.

The legend in the next tables is as follows. The first three columns give the number of vertices $|V|$, the number of arcs $|A|$, and the optimal solution value OPT for each instance. The remaining columns report, for each solution approach, the number of relaxed subproblems (bb) solved by CPLEX, and the elapsed time (cpu) in seconds required to obtain the optimal solution. For the MIP models, we provide the integrality (gap) in percentage between the linear relaxed solution and the optimal one. The elapsed time (t) in seconds to solve the linear relaxation of each MIP model is also given (for CP, it is the execution time of (CPD-CR), which is the same to the B&B algorithm). For B&B, we also show the average cardinality ($\bar{c}$) of the cycles broken during the execution of the algorithm. The symbol "NA" means values not available due to time limit exceeded, whereas "OM" means that CPLEX runs out of memory.

For the benchmark instances of Haouari *et al.* (2013) in Table 1, (CPD) obtains optimal solutions in smaller CPU times for 21 digraphs while CP and B&B get it for 11 and 2 digraphs, respectively. Nevertheless, the average CPU time presented by CP is 1.30 seconds against 1.82 seconds of model (CPD) and 8.50 seconds of B&B. Observe also that the average number of subproblems solved by CP (7.87) is much smaller than those of (CPD) and B&B (180.50 and 881.84, respectively). Model (HMM-RLT) finds all optimal solutions in average CPU time of 13.45 seconds while (IMM) is not able to solve 17 instances to optimality. For digraphs with up to 600 vertices, (IMM) produces tighter LP relaxations than (HMM-RLT) and (CPD). Considering all instances, the average gap of (CPD) is a bit smaller than that one of (HMM-RLT). The average cardinality of broken cycles is 28.45.

Table 1 – Results for benchmark instances of Haouari *et al.* (2013).

Instance			(IMM)				(HMM-RLT)				(CPD)				B&B			CP		
$\|V\|$	$\|A\|$	OPT	bb	cpu	gap	t	bb	cpu	gap	t	bb	cpu	gap	t	$\bar{c}$	bb	cpu	bb	cpu	t
10	29	−185	0	0.15	0.00	0.10	0	0.10	0.00	0.10	0	0.09	0.00	0.10	0.00	0	0.09	0	0.09	0.08
10	29	−232	0	0.14	0.00	0.01	0	0.10	0.00	0.01	0	0.01	0.00	0.01	0.00	0	0.00	0	0.01	0.00
20	70	−392	0	0.19	0.00	0.03	0	0.14	0.00	0.01	0	0.01	0.00	0.01	0.00	0	0.01	0	0.01	0.01
20	74	−446	0	0.19	0.00	0.02	0	0.13	0.00	0.01	0	0.02	0.00	0.01	8.00	8	0.10	1	0.02	0.00
30	110	−744	0	0.56	0.00	0.08	0	0.16	0.00	0.02	0	0.05	0.00	0.01	0.00	61	0.15	4	0.07	0.01
30	114	−815	0	0.27	0.86	0.13	0	0.12	2.33	0.01	0	0.01	1.10	0.01	10.17	0	0.10	0	0.01	0.01
40	154	−1057	0	1.06	0.19	0.28	0	0.25	3.88	0.02	0	0.06	2.37	0.01	6.44	58	0.14	3	0.11	0.01
40	158	−928	0	0.99	1.08	0.22	0	0.27	7.54	0.02	0	0.08	5.93	0.01	4.33	65	0.14	5	0.12	0.00
50	193	−1214	0	1.86	0.99	0.42	0	0.24	1.48	0.02	178	0.09	1.57	0.01	17.93	184	0.20	11	0.37	0.00
50	196	−1281	0	2.39	1.72	0.62	0	0.26	2.34	0.02	0	0.07	2.03	0.01	13.14	251	0.23	11	0.54	0.01
60	234	−1623	14	8.59	0.80	1.01	0	0.28	1.48	0.02	155	0.17	1.79	0.01	11.33	887	0.63	22	1.08	0.01
60	242	−1519	0	5.72	0.86	0.75	0	0.23	2.63	0.02	171	0.12	3.56	0.01	18.87	102	0.22	4	0.15	0.01
70	276	−1824	0	7.24	0.93	1.49	0	0.27	1.75	0.04	180	0.18	1.75	0.01	11.78	106	0.19	5	0.20	0.00
70	285	−1912	0	5.30	0.21	1.12	0	0.22	0.63	0.03	0	0.06	0.63	0.01	3.00	6	0.12	2	0.05	0.00
80	318	−2176	26	38.57	0.18	1.65	0	0.32	0.60	0.02	388	0.27	0.55	0.01	5.63	154	0.24	6	0.38	0.00
80	323	−2012	0	14.70	0.80	1.39	0	0.29	1.64	0.03	0	0.12	1.64	0.01	9.63	45	0.19	2	0.07	0.01
90	353	−2339	39	64.65	0.43	2.45	62	0.42	1.24	0.03	0	0.26	1.24	0.01	11.93	167	0.27	7	0.43	0.00
90	357	−2293	5	23.05	0.35	1.97	0	0.27	0.44	0.04	0	0.09	0.39	0.01	20.00	60	0.22	3	0.13	0.01
100	407	−2628	21	69.94	0.38	2.83	129	0.56	0.42	0.04	0	0.12	0.42	0.01	36.25	164	0.28	5	0.21	0.01
100	414	−2768	12	77.01	0.33	3.12	0	0.29	0.40	0.04	0	0.12	0.43	0.02	32.80	290	0.42	8	0.29	0.01
200	821	−5133	25	1435.22	0.12	33.51	0	0.76	0.27	0.15	181	0.39	0.27	0.02	10.89	225	0.58	5	0.79	0.01
200	827	−5288	NA	NA	0.17	NA	0	1.28	0.47	0.16	0	0.32	0.55	0.02	18.75	207	0.57	8	1.06	0.01
300	1227	−7856	NA	NA	0.09	NA	9	1.29	0.48	0.32	0	0.33	0.10	0.03	60.25	241	1.02	3	0.37	0.02
300	1229	−7988	NA	NA	0.11	NA	134	2.28	0.26	0.44	263	0.72	0.14	0.04	28.16	901	2.69	13	1.41	0.02
400	1657	−10327	NA	NA	0.06	NA	685	7.47	0.10	0.50	0	0.56	0.10	0.06	27.00	243	1.28	6	1.04	0.02
400	1660	−10276	NA	NA	0.00	NA	0	4.19	1.07	0.47	0	0.27	0.50	0.05	2.80	14	0.31	1	0.19	0.03
500	2078	−12999	NA	NA	0.01	NA	63	4.63	0.13	0.62	349	2.75	0.13	0.09	23.50	188	1.22	4	0.47	0.03
500	2085	−13086	NA	NA	0.09	NA	493	6.12	0.16	0.72	121	1.48	0.09	0.09	54.63	1311	6.72	14	2.58	0.03
600	2469	−15573	NA	NA	0.03	NA	865	25.57	0.07	1.00	890	4.84	0.07	0.13	84.63	9225	57.43	32	8.97	0.03
600	2486	−15169	NA	NA	0.06	NA	228	6.60	0.09	1.02	497	2.83	0.09	0.12	55.00	770	5.19	16	2.93	0.03
700	2907	−18188		OM			569	22.31	0.09	1.61	1579	11.15	0.09	0.13	94.05	5925	46.55	21	4.64	0.06
700	2932	−17841		OM			2275	48.08	0.15	1.67	781	5.70	0.05	0.14	29.97	1169	9.56	11	2.57	0.04
800	3280	−20510		OM			1432	18.38	0.11	1.87	49	2.80	0.11	0.19	83.32	2083	30.46	10	2.26	0.05
800	3329	−20269		OM			746	60.44	0.04	1.65	430	6.11	0.04	0.18	97.88	3132	48.44	17	4.12	0.05
900	3693	−23049		OM			679	62.56	0.05	2.28	404	5.68	0.05	0.16	74.69	2913	54.11	21	7.05	0.08
900	3712	−23719		OM			509	86.72	0.03	2.19	58	4.72	0.03	0.28	15.40	77	1.88	3	0.61	0.07
1000	4158	−25680		OM			923	76.46	0.08	2.70	75	6.55	0.08	0.18	17.22	155	3.88	3	1.20	0.07
1000	4176	−26076		OM			297	71.20	0.11	2.73	110	9.96	0.11	0.27	81.65	2123	47.18	12	2.73	0.07
Average				NA			265.74	13.45	0.86	0.60	180.50	1.82	0.74	0.06	28.45	881.84	8.50	7.87	1.30	0.02

Concerning our randomly generated digraphs in Table 2, B&B presents smaller CPU times for 28 instances while CP gets it for 20 instances. The average number of subproblems solved by them is 108.04 and 6.66, respectively. The remaining 2 instances are solved in smaller CPU times by (CPD), which solves 21.50 subproblems in average. Model (HMM-RLT) finds all optimal solutions in average CPU time of 106.62 seconds while model (IMM) solves to optimality only 11 instances. Taking into account only digraphs with 100 vertices, we note that (IMM) produces LP-relaxed solutions with very small gaps. For the whole set of instances, the average integrality gaps of (CPD) and (HMM-RLT) are 0.02 and 0.03, respectively. The average cardinality of broken cycles is 13.75.

Table 2 – Results for new randomly generated instances.

Instance			(IMM)				(HMM-RLT)				(CPD)				B&B			CP		
$\|V\|$	$\|A\|$	OPT	bb	cpu	gap	t	bb	cpu	gap	t	bb	cpu	gap	t	$\bar{c}$	bb	cpu	bb	cpu	t
100	1899	−3968	18	2358.55	0.10	157.20	0	0.97	0.10	0.46	111	1.01	0.13	0.16	22.33	201	0.77	6	0.79	0.14
100	1913	−4085	0	356.41	0.00	122.84	0	0.32	0.00	0.15	0	0.14	0.00	0.04	0.00	0	0.02	0	0.07	0.03
100	1937	−4118	25	2747.55	0.15	37.26	160	1.45	0.17	0.38	0	0.61	0.17	0.04	12.81	269	0.72	9	1.04	0.02
100	1985	−4158	0	820.37	0.00	234.24	0	1.06	0.02	0.31	0	0.46	0.02	0.03	9.00	36	0.11	3	0.21	0.02
100	1983	−4192	0	727.99	0.02	177.56	0	1.39	0.05	0.33	0	0.67	0.05	0.03	23.00	69	0.28	3	0.39	0.01
100	3878	−4501	3	1933.19	0.04	368.18	0	2.68	0.09	0.29	0	1.97	0.07	0.08	11.38	239	1.72	4	0.64	0.03
100	3778	−4450	NA	NA	0.05	327.26	34	2.47	0.14	0.53	500	2.53	0.09	0.07	24.46	318	2.19	17	2.61	0.02
100	3822	−4518	NA	NA	0.04	438.45	39	2.45	0.04	0.50	0	1.80	0.04	0.07	14.33	172	1.23	11	1.40	0.03
100	3892	−4495	0	3297.42	0.00	249.06	0	1.84	0.04	0.59	0	0.86	0.04	0.06	4.33	13	0.11	4	0.53	0.03
100	3870	−4492	9	1360.11	0.05	377.32	0	2.55	0.07	0.55	99	1.85	0.07	0.06	12.63	101	0.75	5	0.73	0.02
100	5835	−4639	NA	NA	0.00	970.24	0	9.48	0.07	1.06	0	3.29	0.02	0.11	5.75	69	0.70	5	1.01	0.03
100	5772	−4626	NA	NA	0.00	1376.51	0	1.46	0.00	0.86	0	0.83	0.00	0.09	16.00	32	0.34	5	0.61	0.03
100	5804	−4653	0	2379.13	0.00	789.74	0	1.08	0.00	0.97	0	1.26	0.00	0.11	57.75	231	2.37	5	0.58	0.03
100	5797	−4635	0	2095.17	0.00	666.17	0	1.60	0.00	1.09	0	0.69	0.00	0.14	16.67	50	0.54	3	0.21	0.02
100	5830	−4603	NA	NA	0.00	934.38	0	4.20	0.00	1.02	0	1.79	0.00	0.11	3.00	3	0.06	3	0.26	0.03
100	7759	−4724	NA	NA	0.00	1085.23	0	13.03	0.11	1.42	0	3.46	0.04	0.19	8.40	210	2.78	7	1.31	0.04
100	7771	−4685	NA	NA	0.00	1815.06	0	2.90	0.02	1.43	0	1.19	0.00	0.18	31.71	222	2.94	3	0.42	0.03
100	7761	−4692	NA	NA	0.00	1185.25	27	5.47	0.00	1.43	0	2.17	0.00	0.17	14.75	59	0.80	9	1.59	0.03
100	7793	−4709	NA	NA	0.00	1698.77	0	2.34	0.02	1.32	0	1.47	0.00	0.13	0.00	0	0.04	0	0.10	0.05
100	7758	−4688	NA	NA	0.00	1360.60	55	27.69	0.09	1.33	0	3.75	0.02	0.20	10.17	61	0.82	9	1.34	0.04
100	9703	−4718	NA	NA	0.02	1190.41	0	11.80	0.02	2.09	114	11.85	0.02	0.40	11.40	114	1.89	15	3.57	0.05
100	9703	−4741	NA	NA	0.00	2212.06	0	2.43	0.02	1.66	0	2.06	0.00	0.18	8.50	51	0.91	11	1.87	0.04
100	9703	−4725	NA	NA	0.00	918.12	0	4.34	0.00	1.87	32	6.03	0.02	0.27	14.00	42	0.73	8	1.91	0.05
100	9703	−4744	0	3209.33	0.00	1426.20	0	3.05	0.00	1.92	0	6.63	0.00	0.27	6.33	19	0.37	1	0.21	0.05
100	9703	−4729	NA	NA	NA	NA	0	8.35	0.00	2.02	0	5.27	0.00	0.23	5.00	25	0.54	9	2.06	0.05
200	7870	−9022	NA	NA	NA	NA	410	59.85	0.03	1.57	0	2.90	0.03	0.17	15.75	252	4.27	6	1.82	0.04
200	7832	−9036	NA	NA	NA	NA	291	19.94	0.09	1.43	219	2.51	0.03	0.12	22.86	480	7.37	15	3.35	0.06
200	7782	−9040	NA	NA	NA	NA	0	3.31	0.00	1.39	0	1.43	0.00	0.14	2.00	2	0.09	1	0.17	0.05
200	7919	−9072	NA	NA	NA	NA	0	7.35	0.01	1.52	0	1.97	0.01	0.17	28.67	86	1.32	5	0.61	0.04
200	7948	−9130	NA	NA	NA	NA	0	6.66	0.00	1.71	0	0.35	0.00	0.13	4.50	9	0.19	6	0.75	0.04
200	15872	−9468	NA	NA	NA	NA	0	37.46	0.03	3.99	0	6.86	0.03	0.80	14.20	284	8.69	7	1.77	0.09
200	15659	−9429	NA	NA	NA	NA	0	49.38	0.01	3.26	0	3.83	0.01	0.39	9.67	29	0.95	1	0.56	0.07
200	15584	−9475	NA	NA	NA	NA	0	66.41	0.04	3.99	0	3.01	0.00	0.31	3.50	14	0.53	12	3.37	0.08
200	15813	−9446	NA	NA	NA	NA	4	146.96	0.01	3.82	0	10.52	0.01	0.31	16.22	146	4.43	3	0.67	0.08
200	15868	−9432	NA	NA	NA	NA	228	159.73	0.00	3.41	0	12.69	0.00	0.46	4.50	9	0.37	4	1.79	0.09
200	23819	−9567	NA	NA	NA	NA	3	204.43	0.01	6.97	0	5.89	0.00	0.56	6.75	54	2.46	7	3.32	0.15
200	23499	−9567	NA	NA	NA	NA	0	121.72	0.00	6.53	0	6.78	0.00	0.72	16.00	48	2.27	3	1.11	0.14
200	23767	−9574	NA	NA	NA	NA	0	10.94	0.00	6.52	0	12.74	0.00	0.53	11.33	34	1.65	10	3.53	0.13
200	23683	−9567	NA	NA	NA	NA	0	17.36	0.00	5.26	0	21.50	0.00	0.84	27.29	191	8.43	5	3.31	0.12
200	23601	−9564	NA	NA	NA	NA	0	78.22	0.00	8.87	0	9.71	0.00	0.54	18.50	148	6.65	11	4.12	0.12
200	31462	−9642		OM			0	17.52	0.01	10.82	0	11.39	0.00	0.94	0.00	0	0.23	2	1.09	0.19
200	31499	−9657		OM			274	175.72	0.00	14.38	0	51.92	0.01	1.08	15.14	106	6.15	16	10.11	0.17
200	31703	−9637		OM			0	211.82	0.00	14.23	0	9.29	0.00	1.96	25.85	336	19.37	7	6.38	0.21
200	31538	−9643		OM			635	1006.56	0.00	13.86	0	40.85	0.00	2.77	4.00	8	0.66	8	6.18	0.17
200	31490	−9647		OM			7	317.49	0.00	12.65	0	18.71	0.00	0.90	17.00	68	4.04	14	8.08	0.19
200	39403	−9685		OM			0	192.34	0.01	14.96	0	32.24	0.00	3.12	21.50	43	3.21	12	9.29	0.24
200	39403	−9681		OM			0	810.62	0.00	13.61	0	17.41	0.00	3.42	20.17	121	8.40	11	7.79	0.27
200	39403	−9677		OM			5	910.50	0.00	14.12	0	46.06	0.01	6.04	10.69	139	9.66	3	3.43	0.23
200	39403	−9691		OM			0	302.97	0.10	14.25	0	16.06	0.00	3.86	15.78	142	9.79	5	5.57	0.29
200	39403	−9683		OM			19	279.46	0.00	13.63	0	28.92	0.00	3.53	11.75	47	3.54	4	2.78	0.31
Average				NA			43.82	106.62	0.03	4.45	21.50	8.78	0.02	0.75	13.75	108.04	2.77	6.66	2.33	0.09

For the grid digraphs of Ibrahim *et al.* (2015b) in Table 3, model (CPD) yields proven optimal solutions in smaller CPU times for 15 instances while model (HMM-RLT) gets it for the remaining 2 instances. The average time spent by model (IMM) to solve each instance is 1.14 seconds while models (CPD) and (HMM-RLT) require 0.23 and 0.24 seconds, respectively. B&B and CP do not find the optimal solution for one instance within the time limit. Nevertheless,

Table 3 – Results for grid instances of Ibrahim *et al.* (2015b).

Instance			(IMM)				(HMM-RLT)				(CPD)				B&B			CP		
$\|V\|$	$\|A\|$	OPT	bb	cpu	gap	t	bb	cpu	gap	t	bb	cpu	gap	t	$\bar{c}$	bb	cpu	bb	cpu	t
100	180	−82	0	0.54	34.15	0.40	0	0.18	117.07	0.16	0	0.16	123.17	0.11	9.77	1250	0.54	15	0.41	0.09
100	180	−167	0	0.71	3.59	0.32	0	0.23	22.16	0.05	0	0.14	22.75	0.02	6.87	268	0.24	6	0.22	0.01
100	180	−62	0	0.58	12.90	0.34	0	0.19	25.81	0.02	0	0.14	250.00	0.02	9.03	12518	3.69	20	0.70	0.01
100	180	−70	0	0.52	14.29	0.21	0	0.22	168.57	0.03	0	0.13	227.14	0.01	11.33	19120	5.93	25	0.79	0.01
100	180	−95	0	0.47	5.26	0.17	0	0.22	101.05	0.03	0	0.14	101.05	0.01	9.59	4478	1.59	29	0.79	0.00
100	180	−121	0	0.57	5.79	0.20	0	0.19	21.49	0.04	0	0.14	30.58	0.01	5.86	82	0.17	5	0.16	0.00
100	180	−102	0	0.60	13.73	0.21	0	0.22	88.24	0.03	1319	0.24	100.00	0.01	14.24	39888	11.86	161	15.21	0.00
100	180	−192	0	0.65	3.13	0.28	0	0.19	11.98	0.03	0	0.14	28.65	0.01	4.29	266	0.23	6	0.21	0.00
100	180	−107	0	0.59	11.22	0.18	0	0.16	48.60	0.03	0	0.12	49.53	0.01	4.88	78	0.19	3	0.15	0.00
200	342	−59	0	1.49	49.15	0.71	0	0.25	89.83	0.06	0	0.22	593.22	0.02	8.01	24332	13.88	25	1.03	0.01
200	342	−76	0	1.31	6.58	0.63	0	0.24	21.05	0.03	0	0.16	251.32	0.02	5.44	1088	0.87	8	0.27	0.01
200	370	−385	0	2.76	4.94	1.93	0	0.36	31.69	0.26	272	0.25	43.64	0.12	13.51	3634	0.54	23	0.88	0.11
200	370	−201	0	1.64	7.96	1.10	0	0.27	49.25	0.07	0	0.22	80.60	0.02	5.92	9154	0.54	17	1.28	0.03
200	370	−113	0	1.48	30.09	1.14	0	0.26	125.66	0.04	0	0.15	284.07	0.01	7.19	284	0.35	5	0.24	0.02
200	370	−371	0	1.94	2.97	1.18	0	0.28	29.11	0.06	0	0.23	50.67	0.01	6.04	1358	0.54	5	0.64	0.02
200	370	−191	0	1.53	2.09	0.71	0	0.25	27.23	0.04	0	0.18	118.85	0.01	5.84	3382	0.54	6	0.33	0.01
200	370	−126	0	2.09	106.35	1.84	0	0.32	174.60	0.04	9955	1.09	312.70	0.01	NA	NA	NA	NA	NA	0.02
Average			0	1.14	18.48	0.68	0	0.24	67.85	0.06	679.18	0.23	156.94	0.02		NA		NA	NA	0.02

these approaches demonstrate to be more efficient than (IMM) for many instances with respect to CPU times. We observe that model (IMM) exhibits smaller gaps when compared to the other MIP models, showing that its corresponding linear relaxation is tighter. For these instances, the average gap of model (CPD) is much larger than the one of (HMM-RLT). The average time spent by (CPD) to solve the whole set of instances is 0.23 seconds, while (IMM) and (HMM-RLT) need 1.14 and 0.24 seconds, respectively. The average time consumed by (CPD) to solve their linear relaxations is 0.02 seconds, while (IMM), (HMM-RLT), and (CPD-CR) (representing the first subproblem of B&B and CP) require 0.68, 0.06, and 0.02 seconds, respectively.

In Tables 4 and 5, for the instances of Taccari (2016), (CPD) finds optimal solutions in smaller CPU times for 31 digraphs while CP and B&B find optimal ones for 8 and 3 digraphs, respectively. (HMM-RLT) and (IMM) present better CPU times for 31 and 17 instances, respectively. The average number of subproblems solved by (IMM) is 2 and the average gap is 8.20%. The average time of (HMM-RLT) is 1.04 seconds and (CPD) requires 3.59 seconds.

Table 4 – Results for instances reported by Taccari (2016).

Instance			(IMM)				(HMM-RLT)				(CPD)				B&B			CP		
$\|V\|$	$\|A\|$	OPT	bb	cpu	gap	t	bb	cpu	gap	t	bb	cpu	gap	t	$\bar{c}$	bb	cpu	bb	cpu	t
27	651	$\star$	0	0.97	0.00	1.07	0	0.21	0.00	0.04	0	0.11	0.00	0.01	3.56	612	0.68	20	0.66	0.12
27	651	$\star$	0	2.60	0.00	1.17	0	0.37	0.00	0.03	0	0.09	0.00	0.00	3.01	56270	49.48	17	0.52	0.01
27	651	$\star$	0	1.06	0.00	0.81	0	0.25	0.00	0.04	0	0.12	0.00	0.00	2.31	83	0.08	5	0.13	0.01
27	651	$\star$	0	0.85	0.00	0.94	0	0.44	0.18	0.03	0	0.17	0.18	0.00	2.41	3724	3.21	3	0.23	0.01
27	651	$\star$	0	1.50	0.00	0.90	0	0.32	0.00	0.03	0	0.26	0.00	0.01	2.23	690	0.61	5	0.23	0.01
27	651	$\star$	0	4.38	0.00	1.35	0	0.26	0.00	0.04	72	0.45	0.00	0.01	3.71	3859	3.32	18	0.66	0.01
27	651	$\star$	0	1.35	0.00	1.16	0	0.34	0.00	0.03	0	0.10	0.00	0.01	3.22	40927	36.32	23	0.67	0.01
27	651	$\star$	0	1.57	0.00	1.02	108	0.69	0.00	0.03	0	0.09	0.00	0.01	3.53	22411	19.42	8	0.18	0.01
27	651	$\star$	0	3.73	0.00	0.96	0	0.28	0.00	0.04	0	0.11	0.00	0.01	4.51	4178	3.68	9	0.29	0.01
27	651	$\star$	0	1.45	0.00	1.20	61	0.50	0.00	0.03	211	0.29	0.00	0.01	3.61	2502	2.21	19	0.64	0.01
27	651	$\star$	0	3.24	0.00	1.12	0	0.22	0.00	0.03	0	0.10	0.00	0.00	3.01	2340	2.05	8	0.24	0.01
27	651	$\star$	0	4.72	0.00	1.01	36	0.83	0.00	0.05	0	0.10	0.00	0.01	2.28	326	0.29	4	0.09	0.01
27	651	$\star$	0	6.73	0.00	1.18	150	0.64	0.00	0.03	263	0.37	0.00	0.00	2.87	11023	9.86	18	0.68	0.01
27	651	$\star$	0	0.74	0.00	0.90	0	0.28	0.00	0.02	0	0.10	0.00	0.00	2.63	121	0.10	5	0.13	0.01
27	651	$\star$	0	0.85	0.00	1.07	123	0.57	0.00	0.03	0	0.10	0.00	0.00	2.83	18125	16.13	8	0.28	0.01
27	651	$\star$	28	0.33	0.00	1.26	0	0.30	0.00	0.03	0	0.10	0.00	0.00	8.06	2298	2.05	4	0.06	0.01
27	651	$\star$	0	3.17	0.00	1.16	3913	3.48	0.27	0.04	9192	3.03	0.40	0.00	NA	NA	NA	88	22.18	0.01
27	651	$\star$	0	3.62	0.00	0.88	0	0.61	0.00	0.04	0	0.10	0.00	0.00	11.91	136534	123.37	6	0.15	0.01
27	651	$\star$	0	1.03	0.00	0.87	0	0.31	0.00	0.03	0	0.11	0.00	0.00	2.79	732	0.64	30	0.82	0.01
27	651	$\star$	0	1.08	0.00	1.13	30	0.55	0.00	0.03	18	0.30	0.00	0.00	2.16	1710	1.45	3	0.08	0.01
27	651	$\star$	0	0.96	0.00	0.68	12	0.35	0.00	0.03	24	0.23	0.00	0.00	2.23	134	0.12	8	0.18	0.01
27	651	$\star$	0	4.74	0.00	1.18	0	0.39	0.00	0.03	0	0.16	0.00	0.00	2.62	1211	1.04	11	0.33	0.01
27	651	$\star$	0	1.13	0.00	1.08	0	0.16	0.00	0.03	0	0.15	0.00	0.00	2.51	339	0.29	7	0.19	0.01
27	651	$\star$	0	1.30	0.00	0.75	6	0.40	0.00	0.03	325	0.47	0.00	0.01	3.19	997	0.87	11	0.28	0.01
27	651	$\star$	0	1.41	0.00	1.00	0	0.55	0.00	0.03	0	0.10	0.00	0.00	2.50	6779	5.85	9	0.22	0.01
27	651	$\star$	0	2.77	0.00	0.96	0	0.24	0.00	0.03	10	0.24	0.00	0.00	2.95	1125	1.00	19	0.54	0.01
27	651	$\star$	0	1.28	0.00	1.11	904	1.43	0.09	0.04	4364	0.79	0.20	0.00	NA	NA	NA	442	103.56	0.00
27	651	$\star$	0	6.51	0.00	1.08	0	0.25	0.00	0.05	64	0.26	0.00	0.00	2.97	8335	7.37	19	0.59	0.01
27	651	$\star$	0	1.28	0.00	1.29	0	0.30	0.00	0.03	0	0.27	0.00	0.01	4.69	2910	2.58	30	0.96	0.00
27	651	$\star$	0	3.14	0.00	0.84	0	0.27	0.00	0.02	96	0.44	0.00	0.00	3.73	20322	17.89	6	0.18	0.01
27	651	-2045	0	1.19	3.18	0.82	761	0.62	454.32	0.03	8755	1.64	427.87	0.01	NA	NA	NA	530	352.49	0.15
27	651	-2356.73	0	2.50	13.15	1.35	2246	0.81	475.37	0.05	20176	3.48	365.97	0.00	NA	NA	NA	99	35.16	0.02
27	651	-202.4	0	0.88	0.00	0.92	395	0.52	5168.62	0.05	1531	0.35	3563.75	0.00	3.75	121510	97.80	14	1.35	0.01
27	651	-730	0	1.05	0.00	0.76	75	0.51	1544.03	0.04	361	0.25	1037.73	0.00	2.72	794488	650.99	42	3.33	0.01
27	651	-5940	0	0.72	0.00	1.10	0	0.31	105.76	0.04	0	0.24	68.21	0.00	2.48	1870	1.50	11	0.73	0.01
27	651	-3420	97	26.30	17.07	0.85	256	0.51	349.15	0.03	4679	0.73	286.92	0.00	3.01	178692	144.98	133	25.77	0.01
27	651	-4985	0	0.78	0.00	1.14	2687	1.21	271.32	0.03	36964	4.48	241.27	0.00	NA	NA	NA	95	67.13	0.01
27	651	-2895.88	0	1.56	5.01	1.21	308	0.50	319.70	0.03	4637	0.66	318.95	0.00	3.42	2576330	2755.46	40	5.97	0.01
27	651	-2870	0	0.86	0.00	0.92	743	0.57	262.58	0.04	4876	1.13	282.58	0.00	NA	NA	NA	34	5.86	0.01
27	651	-154	0	0.81	0.00	0.67	0	0.34	8107.29	0.03	1223	0.43	6732.47	0.00	3.34	1576933	1407.92	27	2.59	0.01
27	651	-3325	0	0.93	0.00	0.81	0	0.52	286.29	0.04	1983	0.47	246.62	0.00	3.03	296102	241.11	18	1.77	0.01
27	651	-1060	0	1.17	0.00	1.02	827	0.68	835.83	0.03	4048	0.70	643.87	0.00	2.87	420554	359.88	31	3.83	0.01
27	651	-618.75	0	2.70	37.27	0.80	1477	0.70	1996.19	0.04	6693	0.79	1906.77	0.00	NA	NA	NA	48	6.46	0.01
27	651	-1334.7	0	0.86	0.00	0.70	302	0.67	569.06	0.03	13368	1.96	513.17	0.00	NA	NA	NA	137	56.03	0.01
27	651	-1030.02	0	0.87	0.00	1.08	4618	0.98	1586.03	0.03	26799	5.16	1572.38	0.00	NA	NA	NA	214	263.49	0.01
27	651	-1472	0	5.22	61.14	1.02	475	0.43	560.15	0.04	26376	3.63	383.56	0.00	NA	NA	NA	NA	NA	0.01
27	651	-1980	0	0.96	0.00	1.04	0	0.44	687.95	0.03	8794	1.57	520.71	0.00	NA	NA	NA	28	4.90	0.01
27	651	-4830	0	3.00	1.45	0.94	0	0.58	186.87	0.04	656	0.42	158.70	0.00	2.80	24437	21.03	48	4.17	0.01
27	651	-4869.95	0	1.16	0.00	0.97	5509	1.42	181.20	0.03	9597	2.20	199.68	0.00	NA	NA	NA	52	22.19	0.01
27	651	-3190	0	1.47	0.84	0.87	247	0.40	485.08	0.03	1774	0.43	309.32	0.00	2.76	394617	326.07	32	3.38	0.01

$\star$ Very small values $\approx -2.6 \times 10^9$

Globally, for the benchmark digraphs, (CPD), B&B and CP outperform existing MIP models for the SPNC. Considering only our randomly generated digraphs, B&B and CP show to be competitive approaches, performing very well for them. For grid and pricing digraphs, (CPD) again shows its efficiency, despite presenting larger integrality gaps when compared to (IMM) and (HMM-RLT). For some instances, B&B and CP do not find optimal solutions.

Table 5 – Results for instances reported by Taccari (2016) (cont.).

Instance			(IMM)				(HMM-RLT)				(CPD)				B&B			CP		
$\|V\|$	$\|A\|$	OPT	bb	cpu	gap	t	bb	cpu	gap	t	bb	cpu	gap	t	$\bar{c}$	bb	cpu	bb	cpu	t
27	651	−4440	0	0.87	0.00	0.92	0	0.34	177.90	0.03	4173	0.54	145.27	0.00	3.20	123804	100.18	35	3.51	0.01
27	651	−4120	0	2.10	3.70	0.95	0	0.37	195.16	0.03	1169	0.65	126.82	0.00	2.63	9868	7.92	7	0.43	0.01
27	651	−1736.7	0	1.54	2.11	1.23	725	0.47	455.89	0.04	3049	0.57	375.04	0.00	NA	NA	NA	30	3.81	0.01
27	651	0	0	1.56	100.00	0.91	0	0.37	100.00	0.04	2391	0.51	100.00	0.00	NA	NA	NA	49	4.99	0.01
27	651	−5839.98	0	0.75	0.00	0.77	0	0.35	299.09	0.04	0	0.23	185.40	0.00	2.57	30854	24.3	4	0.19	0.01
27	651	−1977.46	0	1.56	1.77	0.93	3161	1.02	446.29	0.04	6937	0.85	444.81	0.00	NA	NA	NA	51	6.08	0.01
27	651	−990	0	1.33	0.00	1.18	0	0.47	793.50	0.03	384649	49.66	868.94	0.00	NA	NA	NA	NA	NA	0.00
27	651	−1564	0	0.98	0.00	1.03	455	0.56	695.72	0.03	3157	0.60	672.54	0.00	2.94	118060	93.71	15	1.62	0.01
27	651	−3753.78	0	2.82	1.68	0.88	545	0.54	200.03	0.04	6702	1.09	177.88	0.00	NA	NA	NA	161	44.16	0.01
27	651	−183.21	0	2.48	430.25	0.91	1205	0.84	6488.82	0.03	11022	1.68	5118.07	0.00	NA	NA	NA	106	25.87	0.01
27	651	−2245	0	1.30	0.00	1.07	5522	1.95	406.56	0.03	180805	39.97	382.41	0.00	NA	NA	NA	NA	NA	0.12
27	651	−3163.4	0	1.08	0.00	1.40	3588	3.56	330.25	0.04	8487	5.51	247.15	0.00	NA	NA	NA	57	24.94	0.01
27	651	−1953.6	0	1.37	0.00	1.08	150	1.49	488.17	0.06	5345	1.03	313.64	0.00	3.38	140137	112.56	12	1.84	0.01
27	651	−2655.42	0	1.07	0.00	1.14	0	0.74	385.21	0.03	3928	3.97	256.06	0.00	2.83	586308	515.77	25	5.50	0.01
27	651	−6000	0	0.94	0.00	0.90	57	0.87	107.93	0.03	1199	0.55	83.04	0.00	2.65	13607	10.96	16	1.41	0.01
27	651	−5340	0	4.47	2.76	0.98	2100	1.70	196.14	0.03	2968	0.71	148.08	0.00	3.01	89157	73.44	19	2.83	0.01
27	651	−6340	0	1.00	0.00	1.30	3447	2.20	194.13	0.04	25771	8.38	168.34	0.01	NA	NA	NA	NA	NA	0.01
27	651	−3137.92	0	0.86	0.00	1.29	1638	1.40	288.64	0.03	7059	4.09	286.63	0.01	NA	NA	NA	57	20.51	0.01
27	651	−2870	0	2.91	0.00	0.99	1011	1.71	262.58	0.03	11242	4.47	282.58	0.00	NA	NA	NA	271	381.10	0.01
27	651	−2080	0	0.79	0.00	1.27	0	0.58	542.42	0.03	2747	0.68	425.72	0.00	3.88	2040823	3067.84	28	4.03	0.01
27	651	−5560	0	1.47	0.00	1.19	1046	1.65	192.77	0.03	4056	2.81	142.90	0.00	NA	NA	NA	26	7.50	0.01
27	651	−1060	0	1.83	0.00	1.12	618	1.42	836.75	0.03	5371	1.10	660.38	0.00	NA	NA	NA	34	6.26	0.01
27	651	−1460	0	5.21	1.03	1.03	2732	2.41	801.53	0.05	10860	3.37	750.47	0.00	NA	NA	NA	NA	NA	0.01
27	651	−1822.62	0	1.48	0.00	1.14	4962	3.24	397.22	0.03	72045	17.32	352.87	0.01	NA	NA	NA	NA	NA	0.01
27	651	−3001.7	0	1.63	0.00	1.26	3468	3.10	483.85	0.03	72831	17.68	473.87	0.00	NA	NA	NA	NA	NA	0.01
27	651	−1536	71	8.07	54.43	0.99	477	0.89	539.54	0.03	52318	8.04	363.41	0.00	NA	NA	NA	NA	NA	0.01
27	651	−4930	0	4.14	0.61	1.07	1775	2.46	217.80	0.03	43723	14.99	154.67	0.00	NA	NA	NA	180	198.00	0.01
27	651	−5150	0	1.29	0.00	1.26	141	0.91	174.81	0.04	1449	0.68	142.62	0.00	3.72	39096	32.96	72	11.59	0.01
27	651	−5139.97	0	1.25	0.00	1.09	5593	5.04	168.91	0.03	13465	4.89	183.94	0.00	NA	NA	NA	58	59.83	0.01
27	651	−5440	0	1.37	0.00	1.02	0	0.53	249.88	0.04	0	0.48	140.22	0.00	3.44	28569	24.09	5	0.41	0.01
27	651	−5260	0	1.42	0.00	1.12	0	1.10	140.72	0.04	4166	1.84	107.03	0.00	3.51	75967	63.37	27	4.57	0.01
27	651	−4400	0	4.99	0.68	0.99	6315	2.65	180.80	0.03	6131	2.66	135.45	0.00	3.13	775850	679.61	17	4.27	0.01
27	651	−4716.74	0	1.40	0.00	1.29	3567	2.25	185.37	0.03	9141	4.21	153.88	0.00	NA	NA	NA	28	12.62	0.01
27	651	−2090	0	1.31	0.00	1.19	1622	1.66	623.93	0.03	71899	18.50	527.15	0.01	NA	NA	NA	NA	NA	0.01
27	651	−7019.96	0	0.78	0.00	0.94	90	0.88	233.68	0.03	2247	0.59	149.25	0.00	3.27	138239	114.55	8	0.69	0.01
27	651	−2317.43	0	1.30	0.00	1.21	3510	4.85	369.02	0.04	16943	3.99	364.88	0.00	NA	NA	NA	83	29.55	0.01
27	651	−3060	0	1.45	0.00	1.07	119	1.73	191.40	0.03	123249	27.61	214.30	0.00	NA	NA	NA	NA	NA	0.01
27	651	−2344	0	1.45	0.00	1.14	33	0.89	438.20	0.04	6553	2.57	416.04	0.00	3.42	762044	677.82	253	91.37	0.00
27	651	−4126.93	0	1.20	0.00	1.18	665	1.47	178.83	0.03	60752	16.14	152.76	0.00	NA	NA	NA	NA	NA	0.01
27	651	−1915.63	0	1.20	0.00	1.16	3699	4.23	533.41	0.04	38381	8.20	405.70	0.00	NA	NA	NA	87	65.73	0.01
Average			2.18	2.23	8.20	1.04	1003.39	1.04	495.85	0.03	16514.54	3.59	406.46	0.00		NA		NA	NA	0.01

As we can see in our numerical experiments, although instances with large gaps (see Tables 3, 4 and 5) lead to poor execution-time performances of both B&B and CP algorithms, we find no strong evidence to classify instances being easy or difficult to solve w.r.t. small or large integrality gaps. Indeed, consider the numerical results reported in Table 6 for the 9 instances modified according to Proposition 4. They contain the structure depicted in Figure 16. By analyzing such table, we note that in general both B&B and CP present better CPU times than those obtained by models (HMM-RLT) and (CPD) for these instances presenting very large integrality gaps. Therefore, it seems that presenting small or large integrality gaps is not conclusive to make instances easy of difficult to deal with the proposed solution approaches.

Table 6 – Results for extended digraphs.

Instance			(HMM-RLT)				(CPD)				B&B			CP		
$\|V\|$	$\|A\|$	OPT	bb	cpu	gap	t	bb	cpu	gap	t	$\bar{c}$	bb	cpu	bb	cpu	t
903	3698	−23049	628	89.71	258.34	3.33	639	10.66	260.08	0.45	83.66	2928	55.16	22	7.79	0.22
903	3717	−23719	116	36.40	251.03	4.17	152	8.83	252.72	0.46	10.75	86	1.90	3	1.89	0.24
1003	4163	−25680	227	109.75	232.09	6.09	117	7.01	233.49	0.44	8.04	201	5.05	3	1.30	0.24
1003	4181	−26076	1775	284.10	228.59	5.40	0	6.26	229.98	0.61	87.96	2111	46.51	11	3.05	0.23
203	39408	−9685	0	11.50	597.93	9.61	0	39.80	616.46	3.40	26.14	183	12.55	5	3.96	0.56
203	39408	−9681	0	475.02	598.18	9.33	0	32.47	616.72	3.54	4.23	55	3.81	5	4.27	0.36
203	39408	−9677	8	683.53	598.44	8.69	0	34.88	616.98	5.30	9.87	79	5.28	3	3.50	0.30
203	39408	−9691	0	10.96	597.56	9.27	0	27.54	616.08	3.31	10.00	80	5.43	5	4.48	0.30
203	39408	−9683	0	11.24	598.06	9.10	0	87.58	616.59	3.16	4.28	77	5.19	1	1.33	0.31

2.7 Conclusion

We propose new solution approaches for the shortest path with negative cycles, including a compact MTZ-primal-dual model (CPD), a combinatorial B&B algorithm, and a cutting-plane (CP) method. Extensive numerical experiments show that they outperform existing solution techniques for this problem (IBRAHIM *et al.*, 2009; HAOUARI *et al.*, 2013) while obtaining optimal solutions for all instances in a few seconds (for B&B and CP, some instances are not solved to optimality). For all benchmark (HAOUARI *et al.*, 2013) and randomly generated instances, model (CPD) alternates better results with both B&B and CP algorithms. Concerning the grid instances (IBRAHIM *et al.*, 2015b), model (CPD) globally shows to be more attractive than the remaining solution strategies, despite its linear relaxation presents larger integrality gaps when compared to the ones obtained by the other models. Finally, for pricing subproblems (TACCARI, 2016), (CPD), (HMM-RLT), and (IMM) performed better for them. There was no strong evidence to conclude if large integrality gaps play an important role in the instance difficulty for these models. Further work in this direction is needed to characterize challenging instances of this problem.

3 CONSTRAINED SHORTEST PATH TOUR PROBLEM

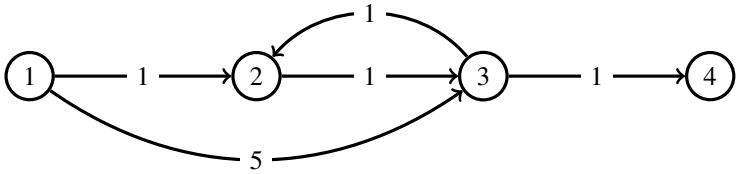
We devote this chapter to the study of the constrained shortest path tour problem (CSPTP). Our pilot research was presented at the Joint EURO/ALIO International Conference 2018 on Applied Combinatorial Optimization (EURO/ALIO 2018). A complete version of this study has been submitted to the International Transactions in Operational Research (SARAIVA; ANDRADE, 2019).

3.1 Introduction

Let $D = (V,A)$ be a connected digraph where V is the set of nodes and A is the set of arcs. Let c_{ij} be a non-negative cost assigned to every arc $(i, j) \in A$. Given two distinct nodes $s,t \in V$, an integer value $N > 1$, and node disjoint subsets $T_i \subset V$ $(i = 1,\ldots,N)$ such that $s \in T_1$ and $t \in T_N$, the CSPTP aims at finding the shortest trail from s to t while successively visiting at least one node in $T_1, T_2, \ldots, T_N$, in this order. The trail starts at $s \in T_1$ and goes to some node in T_2 (through nodes of $V - T_2$), then goes to some node in T_3 (through nodes of $V - T_3$), and so on, finally ending at $t \in T_N$ (through nodes of V). It is worth mentioning that trails may visit any node more than once, but repeating an arc is not allowed.

The CSPTP was introduced by Ferone *et al.* (2016) and can be viewed as a variant of the shortest path tour problem (SPTP) (FESTA, 2012; FESTA *et al.*, 2013). Basically, the difference is that the latter problem allows repeated arcs. We formalize such a distinction considering Figure 20, which depicts a digraph with its arc costs. Let $s = 1$ and $t = 4$ be the source and destination nodes, respectively. Clearly, arcs $(1,2)$, $(2,3)$, and $(3,4)$ represent the standard shortest path from s to t, whose cost is 3. Now, let $T_1 = \{1\}$, $T_2 = \{3\}$, $T_3 = \{2\}$, and $T_4 = \{4\}$ be node disjoint subsets to visit, in this order. The SPTP optimal solution is the sequence of arcs $(1,2)$, $(2,3)$, $(3,2)$, $(2,3)$, and $(3,4)$, whose cost is 5. Note that arc $(2,3)$ is used twice. As the CSPTP does not allow repeated arcs, its corresponding optimal solution is composed of arcs $(1,3)$, $(3,2)$, $(2,3)$, and $(3,4)$, in this order, whose cost is 8.

Figure 20 – Example of a directed graph.



Source: Ferone *et al.* (2016).

Real-world CSPTP applications typically arise in cargo transportation operations in which each customer has a set of available warehouses to where goods can be delivered. Goods are stored in a container to be transported and delivered to customers following a priority queue according to the order of client visits. As observed by Ferone *et al.* (2016), the CSPTP can also be viewed as a special case of the network interdiction problem on a flow network (LIM; SMITH, 2007) in which an attacker disables all the arcs of a network whenever they are used to ship flow, with the aim of minimizing the net profit that can be obtained from shipping a commodity across the network. A profit can be associated with nodes or arcs. Whenever the profit of an arc is collected, it is forbidden to traverse it again since it is disrupted after traveling on it for the first time. In this context, a special case of the network interdiction modeled by the CSPTP occurs, when a profit is associated with all the subsets of a given sequence that must be visited in a fixed order. Another application concerns reliable network design (HEEGAARD; TRIVEDI, 2009), where one selects ways to travel arcs whose usage does not compromise the survivability of the flow that must be sent along the network through ordered sets of nodes that must be crossed at least once (FERONE *et al.*, 2016).

Theoretical properties of the CSPTP are discussed by Ferone *et al.* (2016), who show how the Hamiltonian path problem (HPP) (BERTOSSI, 1981) is polynomially Karp-reducible to the CSPTP. It is also thanks to Ferone *et al.* (2016) the first solution approaches: an exact branch-and-bound (B&B) method and a greedy randomized adaptive search procedure (GRASP). In the B&B approach, the CSPTP is reduced to the path avoiding forbidden pairs problem (PAFPP) (SRIMANI; SINHA, 1982) on an auxiliar multistage digraph. If the solution of a B&B subproblem does not contain a forbidden pair, then a CSPTP feasible solution is known. Otherwise, new subproblems are generated by branching on some forbidden pair. It is worth mentioning that their B&B is able to efficiently solve only small-size instances. When dealing with larger ones, their GRASP procedure, which operates with constructive and local search phases based on a classical shortest path algorithm (DIJKSTRA, 1959), is used to obtain feasible solutions. Recently, Ferone *et al.* (2019) conceived an alternative B&B approach whose subproblems are SPTPs, which, in turn, are solved by dynamic programming. If the resulting tour does not contain repeated arcs, then it is feasible for the CSPTP. Otherwise, the algorithm branches on a given repeated arc.

B&B can take advantage of valid combinatorial optimization problem relaxation to solve a given integer programming (IP) problem. Indeed, the CSPTP can be addressed by

means of another problem structure, e.g., PAFPP (FERONE *et al.*, 2016) and SPTP (FERONE *et al.*, 2019). Alternatively, one can explore specific IP models for the CSPTP. Andrade e Saraiva (2018), for instance, recently proposed a dummy node-based model that showed to be very efficient.

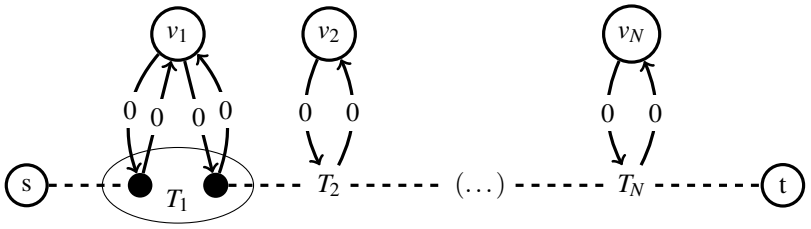
In this chapter, we extend the ideas of Andrade e Saraiva (2018) and extensively evaluate the dummy node-based model for a set of 321 instances. We perform comparative analysis between the dummy node-based model and a new one, called frontier node-based model, that we independently developed in parallel with a similar model recently reported by Ferone *et al.* (2019). Moreover, we propose new valid inequalities that enhance the computational performance of the latter model, as well as explore the coefficient matrix of its linear relaxation to design heuristics embedded into a Lagrangian-based framework. At each iteration of the dual ascent procedure, we use the information obtained from the Lagrangian problem to find promising feasible solutions. Experiments carried out on benchmark data sets from the literature (FERONE *et al.*, 2016) show the efficiency of the valid inequalities and the effectiveness of our Lagrangian framework. Computational results show, in fact, that these data sets do not require a great computational effort of IP models as their optimality is always reached in the root node of the CPLEX branch-and-cut search tree. This motivated us to generate two new challenging data sets for the problem. We show that state-of-the-art B&B algorithms fail to solve all new instances due to memory limitation, while the existing GRASP heuristic does not achieve any optimal solution for them. On the other hand, our solution approaches find no difficulty in handling all the new instances to optimality.

The remainder of the chapter is organized as follows. Section 3.2 presents IP models for the CSPTP, remarks on the frontier node-based model, and valid inequalities for the problem. Section 3.3 details the Lagrangian-based framework. Section 3.4 reports experiments on benchmark and new randomly generated data sets. Section 3.5 closes the chapter with a brief conclusion.

3.2 Integer programming models

This section presents two IP models for the CSPTP.

Figure 21 – Representation of the augmented digraph.



Source: The author.

3.2.1 Dummy node-based model

Initially, we modify each CSPTP instance as follows. For every node disjoint subset T_k ($k = 1,\ldots,N$), we create a dummy node v_k and add it to V; for all $u \in T_k$, we add arcs (u, v_k) and (v_k, u) to A and set their costs to zero, as in Figure 21. Refer to V' and A' as the augmented sets of nodes and arcs, respectively.

Let $P = \{(s, v_1), (v_1, v_2), (v_2, v_3), \ldots, (v_N, t)\}$ consist of $N+1$ ordered pairs of nodes and let (a_k, b_k) be the k-th element of P ($k = 1,\ldots,N+1$). The idea behind the dummy node-based model (ANDRADE; SARAIVA, 2018) is to find $N+1$ paths, each one having source a_k and destination b_k, and concatenate them in order to compute the optimal trail from s to t. For this aim, we employ binary decision variables x_{ij}^k that take the value 1 if arc (i, j) belongs to the path from a_k to b_k, and 0 otherwise, for all $(i, j) \in A'$ and for $k = 1,\ldots,N+1$. The model is as follows.

$$\text{(DN)} \quad \min \quad \sum_{k=1}^{N+1} \sum_{(i,j)\in A'} c_{ij} x_{ij}^k \tag{3.1}$$

$$\text{s.t.} \quad \sum_{(i,j)\in A'} x_{ij}^k - \sum_{(j,i)\in A'} x_{ji}^k = \begin{cases} 1, \text{if } i = a_k, \\ -1, \text{if } i = b_k, \\ 0, \text{otherwise.} \end{cases}, \quad k = 1,\ldots,N+1, \forall i \in V' \tag{3.2}$$

$$x_{iv_k}^k - x_{v_k i}^{k+1} = 0, \quad k = 1,\ldots,N, \; \forall\, (i, v_k) \in A' \tag{3.3}$$

$$\sum_{k=1}^{N+1} x_{ij}^k \leq 1, \quad \forall\, (i, j) \in A' \tag{3.4}$$

$$x_{iv_p}^k = 0, \quad \forall\, (i, v_p) \in A', k = 1,\ldots,N+1 : k \neq p \tag{3.5}$$

$$\sum_{(i,s)\in A'} x_{is}^1 = 0 \tag{3.6}$$

$$\sum_{(t,i)\in A'} x_{ti}^{N+1} = 0 \tag{3.7}$$

$$\sum_{(i,j)\in A'} x_{ij}^{k} \leq 1, \quad k = 1,\ldots,N+1, \forall\, j \in V' \tag{3.8}$$

$$x_{ij}^{k} \in \{0,1\}, \quad \forall\,(i,j) \in A', k = 1,\ldots,N+1 \tag{3.9}$$

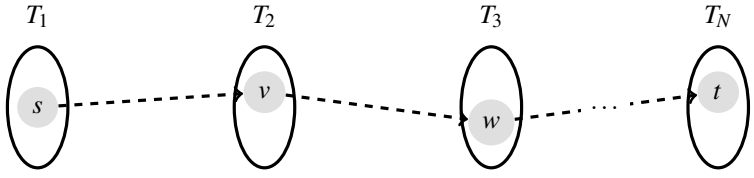
The objective function (3.1) minimizes the cost of the trail from s to t. Constraints (3.2) ensure flow conservation at every node of each path whose origin and destination are elements of P. Constraints (3.3) impose that the last arc belonging to a given path is the first arc belonging to the next subsequent path. Constraints (3.4) guarantee that each arc is used at most once. Constraints (3.5) impose that x variables referred to arcs (i, v_k), with $i \in T_k$, are null in any path $p \neq k$. Constraint (3.6) (resp. (3.7)) states that no arc enters (resp. leaves) the source s (resp. destination t) in the first (resp. last) path. In each path, constraints (3.8) establish that the number of arcs entering any node is at most one. Finally, constraints (3.9) define the domain of the decision variables. Model (DN) contains $O(N \times |A'|)$ binary decision variables and $O(N \times |V'| + |A'|)$ constraints.

3.2.2 Frontier node-based model

Based on model (DN) (ANDRADE; SARAIVA, 2018), it is straightforward to adapt it to obtain an alternative IP model referred to as frontier node-based model with no use of dummy nodes. The idea relies on the observation that the optimal trail can be viewed as a concatenation of $N-1$ paths where the k-th path starts at the frontier node of T_k and ends at the frontier node of T_{k+1} ($k = 1,\ldots,N-1$). A frontier node is an element of T_k ($k = 1,\ldots,N$) that is source and/or destination of a simple path. Thus, any feasible solution contains exactly N frontier nodes, being two of them trivial due to the problem definition, namely, $s \in T_1$ and $t \in T_N$. See Figure 22 for an example.

Working with the concept of frontier node was independently developed in parallel by Ferone *et al.* (2019). Because of this, we reproduce in the sequel exactly the (frontier node-

Figure 22 – Representation of frontier nodes: s is source of the first path; v is destination (resp. source) of the first (resp. second) path; w is destination (resp. source) of the second (resp. third) path; and t is destination of the last path.



Source: The author.

based) model of Ferone *et al.* (2019) and provide some important remarks left open by the authors.

Consider decision variables y_i denoting if i is a frontier node (value 1) or not (value 0), for all $i \in \bigcup_{k=1}^{N} T_k$. Let x_{ij}^k be decision variables indicating if arc (i,j) belongs to the k-th path (value 1) or not (value 0), for all $(i,j) \in A$ and for $k = 1,\ldots,N-1$. The frontier node-based model (FN) is as follows.

$$\text{(FN)} \quad \min \quad \sum_{k=1}^{N-1} \sum_{(i,j)\in A} c_{ij} x_{ij}^k$$

$$\text{s.t.} \quad \sum_{(i,j)\in A} x_{ij}^k - \sum_{(j,i)\in A} x_{ji}^k = \begin{cases} -y_i, \text{if } i \in T_{k+1}, \\ y_i, \text{if } i \in T_k, \\ 0, \text{otherwise.} \end{cases}, \quad k = 1,\ldots,N-1, \forall i \in V \tag{3.10}$$

$$\sum_{i\in T_k} y_i = 1, \quad k = 1,\ldots,N \tag{3.11}$$

$$\sum_{k=1}^{N-1} x_{ij}^k \leq 1, \quad \forall (i,j) \in A \tag{3.12}$$

$$y_i = 1, \quad i \in \{s,t\} \tag{3.13}$$

$$x_{ij}^k \in \{0,1\}, \quad \forall (i,j) \in A, k = 1,\ldots,N-1 \tag{3.14}$$

$$y_i \in \{0,1\}, \quad k = 1,\ldots,N, \forall i \in T_k \tag{3.15}$$

Constraints (3.10) ensure flow conservation at any node belonging to the trail. Constraints (3.11) enforce that each subset T_k has exactly one frontier node. Constraints (3.12) state that each arc can be used at most once. Constraints (3.13) impose s and t to be frontier nodes. Constraints (3.14) and (3.15) define the variable domains. Model (FN) has $O(N \times |A| + \sum_{k=1}^{N} |T_k|)$ binary variables and $O(N \times |V| + |A|)$ constraints.

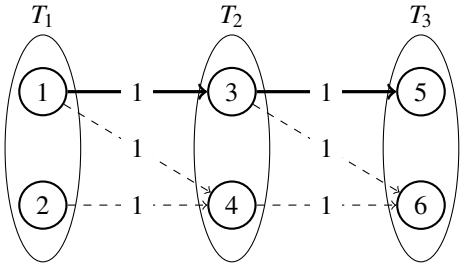
Now, we point out remarks concerning the frontier node-based model (FN) that are relevant from theoretical and practical aspects.

Proposition 6 *The integrality on the y variables in model (FN) is not necessary.*

Proof 6 *We claim that y variables can be continuous, say* $0 \leq y \leq 1$. *This follows from constraints* (3.10) *and the integrality on the x variables* (3.14).

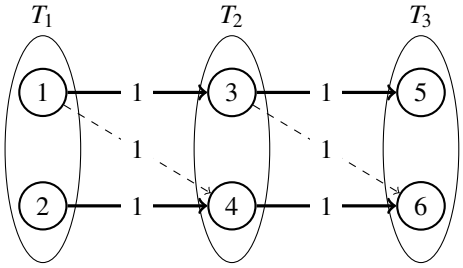
Proposition 7 *Constraints* (3.11) *are not necessary to (FN) be correct.*

Figure 23 – Solution in bold lines with every disjoint subset having exactly one frontier node.



Source: The author.

Figure 24 – Solution in bold lines with every disjoint subset having two frontier nodes.



Source: The author.

Proof 7 *As $y_s = 1$ by* (3.13)*, the flow conservation constraints* (3.10) *ensure that $s \in T_1$ is the source of the first path whose destination must be some node $v \in T_2$, for which $y_v = 1$. Constraints* (3.10) *impose that the second path starts at $v \in T_2$ and ends at some $u \in T_3$, for which $y_u = 1$. This reasoning is applied to every k-th path, $k = 3, \ldots, N-1$. When $k = N-1$, in particular, $y_t = 1$ by* (3.13) *and hence $t \in T_N$ is the destination of the last path. The knapsack constraints* (3.12) *impose that each arc is used at most once. Thus, constraints* (3.11) *are not necessary to model (FN) be correct.*

Despite redundant, constraints (3.11) speed up CPLEX execution time for this model. This is due to the fact that (FN) contains feasible solutions consisting of multiple frontier nodes for each individual subset T_k ($k = 1, \ldots, N$), thus allowing the existence of multiple disjoint trails from T_1 to T_N. These solutions will never be of minimum cost as there is no negative arc cost. An example is shown in Figs. 23 and 24.

Proposition 8 *There is no arc entering (resp. leaving) node s (resp. t) in the first (resp. last) path.*

$$\sum_{(i,s)\in A} x_{is}^{1} = 0 \quad \textit{and} \quad \sum_{(t,i)\in A} x_{ti}^{N-1} = 0 \tag{3.16}$$

Proof 8 *As s (resp. t) is the source (resp. destination) of the first (resp. last) path and arc costs are non-negative, then visiting s (resp. t) more than once in the first (resp. last) path leads to non-optimal solutions.*

Proposition 9 *In each individual path, the number of arcs entering any node $j \in V$ is at most one.*

$$\sum_{(i,j)\in A} x_{ij}^k \leq 1, \quad k = 1,\ldots,N-1, \forall j \in V \tag{3.17}$$

Proof 9 *Visiting an arbitrary node more than once in any path would induce a subtour with non-negative cost. As arc costs are non-negative, the corresponding solution cannot be optimal.*

Proposition 10 *In any k-th path from T_k to T_{k+1} ($k = 1,\ldots,N-2$), the first element visited in T_{k+1} can be considered its frontier node.*

Proof 10 *Consider an optimal solution (x^*, y^*) where the k-th path visits two or more nodes in T_{k+1}. Let u and v be the first and the last nodes, respectively, of T_{k+1} in the k-th path of (x^*, y^*). Node v is the source of the $(k+1)$-th path in this solution. Now consider a new feasible solution $(\bar{x}, \bar{y})$ exactly composed by the same set of arcs of (x^*, y^*) in the following way. Node u replaces v as the frontier node of T_{k+1} in $(\bar{x}, \bar{y})$; the path from u to v of the k-th path and the $(k+1)$-th path from T_{k+1} to T_{k+2} of (x^*, y^*) now constitute the $(k+1)$-th path from T_{k+1} to T_{k+2} in $(\bar{x}, \bar{y})$. All remaining paths and frontier nodes of (x^*, y^*) are kept the same in $(\bar{x}, \bar{y})$. Both solutions refer to the same trail structure. Thus, they have the same value and therefore $(\bar{x}, \bar{y})$ is also optimal.*

Corollary 1 *Any k-th path from T_k to T_{k+1}, for $k \in \{1,\ldots,N-2\}$, ends at the first node of T_{k+1}. Consequently,*

$$x_{ij}^k = 0, \quad \forall\, (i,j) \in A \,|\, i \in T_{k+1},\ k \in \{1,\ldots,N-2\} \tag{3.18}$$

Proof 11 *By Proposition 10, no arc leaves T_{k+1} in the k-th path, $k = 1,\ldots,N-2$.*

In what follows, we refer to (FN)$^+$ as model (FN) with the valid inequalities presented above.

3.3 Lagrangian framework

In model (FN), complicating constraints are the knapsack inequalities (3.12). Thus, if we relax them by means of non-negative Lagrange multipliers $\mu \geq 0$, we obtain the Lagrangian problem

$$L(\mu) \quad \min \quad \sum_{k=1}^{N-1} \sum_{(i,j)\in A} c_{ij} x_{ij}^k + \sum_{(i,j)\in A} \mu_{ij} \Big(\sum_{k=1}^{N-1} x_{ij}^k - 1 \Big) \tag{3.19}$$

s.t. (3.10), (3.11), (3.13) – (3.15)

For any $\mu \geq 0$, $L(\mu)$ is a lower bound on the optimal solution value provided by (FN). The linear relaxation of the Lagrangian problem is integral (see Proposition 11 and Corollary 2). In this case, the Lagrangian dual problem

$$\max_{\mu \geq 0} \quad L(\mu) \tag{3.20}$$

has the same optimal value (GEOFFRION, 1974) of the linear relaxation of (FN). The Lagrangian dual problem is handled by a classical subgradient algorithm as in the self-explaining pseudocode of Algorithm 2. In this algorithm, the function `stopCondition()` is related to some stopping criterion, e.g. number of subgradient iterations. Procedure `subgradient(`$L(\mu^k)$`)` computes the subgradient of $L(\mu^k)$ at $\bar{x}$ having components given by $\sum_{k=1}^{N-1} \bar{x}_{ij}^k - 1$, for all $(i,j) \in A$.

Proposition 11 *The constraint matrix defined by* (3.10) *is totally unimodular (TU).*

Proof 12 *The coefficient matrix associated with the x variables is composed of $N-1$ network flow matrices known to be individually TU. Columns y associated with nodes of T_1 and T_N have only one non-zero entry that is equal to -1 and $+1$, respectively. Columns y associated with each node of T_k ($k = 2,\ldots,N-1$) have exactly two non-zero entries that are equal to $+1$ and -1. The complete matrix fits the well known sufficient condition to a matrix be TU.*

Corollary 2 *The constraint matrix $A_{L(\mu)}$ associated with the Lagrangian problem $L(\mu)$ is TU.*

Proof 13 *As shown by Proposition 11, the constraint matrix defined by* (3.10) *is TU. This is due to the fact that such a matrix is a classical network matrix, which is TU. Now, we want to show that the constraint matrix defined by* (3.10) *and* (3.11) *is also a network matrix. To prove it, we resort to an algorithm for recognizing network matrices (SCHRIJVER, 1998). Note that $A_{L(\mu)}$ has columns with at most three non-zero coefficients (those related to y_i, $i \in \bigcup_{k=2}^{N-1} T_k$). Define, for any row index $i = 1,\ldots,m$ of $A_{L(\mu)}$, an undirected graph G_i as follows. The vertex set of G_i is $\{1,\ldots,m\}\backslash\{i\}$. Two vertices j and k are adjacent in G_i if and only if $A_{L(\mu)}$ has a column that has non-zeros in row positions j and k of and a zero in position i. Then, if $A_{L(\mu)}$ is a network matrix, there exists an i for which G_i is disconnected. Observe that constraints* (3.11) *contain N rows, each one related to a subset T_k ($k = 1,\ldots,N$). Choose a row $i \in \{2,\ldots,N-1\}$. Without loss of generality, suppose $i = 2$. Each set of columns represented by x^k ($k = 1,\ldots,N-1$) leads to a connected component H_k of G_i, since x-columns define $N-1$ independent network flow*

Algorithm 2: Subgradient algorithm.

Data: upper bound UB on the optimal solution value, maximum number of subgradient iterations K, maximum number of iterations β without lower bound (LB) improvement.
Result: LB on the optimal solution value.
$\mu^0 \leftarrow 0$; $\lambda_0 \leftarrow 0.75$; $k \leftarrow 0$; $LB \leftarrow -\infty$; $IterNoImprovedLB \leftarrow 0$;
while (`stopCondition()` $= false$) **do**
    `solve`$(L(\mu^k))$ and let its optimal solution value be L^*;
    $\gamma^k \leftarrow$ `subgradient`$(L(\mu^k))$; // subgradient of $L(\mu^k)$ at x^k
    $\theta_k \leftarrow \lambda_k(UB - L^*)/||\gamma^k||^2$; // step-size
    if $(LB < L^*)$ **then**
        $LB \leftarrow L^*$;
    else
        $IterNoImprovedLB \leftarrow IterNoImprovedLB + 1$;
    end if
    $\mu^{k+1} \leftarrow \max\{0, \mu^k + \theta_k\gamma^k\}$;
    if $(IterNoImprovedLB \ module \ \beta = 0)$ **then**
        $\lambda_{k+1} \leftarrow \lambda_k/2$;
    else
        $\lambda_{k+1} \leftarrow \lambda_k$;
    end if
    $k \leftarrow k + 1$;
end while

matrices, all of these columns with value zero in the i-th row position. Note that component H_1 is not connected with H_2. The columns of $y(T_2)$ have non-zero coefficients in the first and second blocks of network flow matrices related to the first and second subpaths, which define H_1 and H_2. These components are clearly not connected. This is because the columns of $y(T_2)$ have non-zero coefficients in the i-th position. Thus, by construction of G_i, a vertex of H_2 cannot be adjacent to a vertex of H_1 in G_2. In fact, for a general $k = 2, \ldots, N-1$, vertices of component $\bigcup_{i=k}^{N-1} H_i$ are not connected with any vertex of component $\bigcup_{i=1}^{k-1} H_i$. For the example above, H_1 is not connected with $\bigcup_{i=2}^{N-1} H_i$.

In Annexe B, we show the construction of a disconnected graph G_i $(i = 1, \ldots, m)$ when considering a given matrix $A_{L(\mu)}$.

Our Lagrangian relaxation (LR) scheme has ingredients to guide the construction of feasible solutions while extracting information from the Lagrangian multipliers and the solution provided by $L(\mu)$, which can be feasible for the original problem (contains only non-repeated arcs) or infeasible (contains repeated arcs). The former case gives a valid upper bound for the problem when considering original arc costs. The latter case is a CSPTP infeasible solution

for which we attempt to turn it into a feasible one through a repair algorithm (Algorithm 3) that makes successive calls to the classical algorithm of Dijkstra (1959) as explained in the sequel. In a nutshell, the idea behind the repair routine is as follows. Let H be a given optimal solution for $L(\mu)$ consisting of a set of arcs A' and N frontier nodes $f_1, f_2, \ldots, f_N$, each one related to $T_1, T_2, \ldots, T_N$, respectively. We initialize a matrix of arc costs $C_{|A|\times(N-1)}$ whose rows are associated with the arcs and columns are associated with the $N-1$ paths from T_1 to T_2, from T_2 to T_3, and so on, finally ending in T_N. Each entry C_{ak} denotes the cost of arc $a \in A$ to be considered in the k-th path. For every path $k = 1, \ldots, N-1$, we run the Dijkstra's algorithm from f_k to f_{k+1} under $D = (V, A)$ and arc costs $C_{<*,k>}$, i.e., when computing the k-th shortest path, we take into account only arc costs from the k-th column of the aforementioned matrix. To avoid further infeasibilities, arcs in the solution of Dijkstra's algorithm from previous iterations are not allowed to be in subsequent shortest paths. The cost of the feasible solution is the sum of $N-1$ shortest paths considering original arc costs. We work with distinct strategies to initialize matrix C. For the row associated with arc $(i, j) \in A$:

- **Strategy I -** Check in which paths (i, j) is present. Assign the cost 0 to the corresponding columns and set the remaining columns with a big positive constant $\mathscr{M}$. If the arc is not present in any path, keep its original arc cost in all columns.
- **Strategy II -** For every arc $a \in A$, set its cost $C_{ak} \leftarrow c_a + \mu_a$, for $k = 1, \ldots, N-1$, where c_a (resp. μ_a) is the original cost (resp. Lagrangian multiplier) of this arc.
- **Strategy III -** For each arc (i, j) of a Lagrangian solution H, we identify its number of occurrences m_{ij} in this solution. If $m_{ij} > 1$, then we run m_{ij} Dijkstra's calls to obtain m_{ij} arc disjoint simple paths from i to j under Lagrangian arc costs. During this process, we avoid using repeated arcs from previous paths. The cost of the resulting feasible solution considers original arc costs.

Considering the subgradient algorithm and the repair procedures, we propose deterministic and non-deterministic variants of Lagrangian heuristics for the CSPTP. The deterministic Lagrangian heuristic (DLH) works as follows. At each iteration of Algorithm 2, we solve the Lagrangian problem and obtain the subgradient associated with its solution. Then, we run Algorithms 3 and 4 to obtain candidate upper bounds (recall that function `setCost()` in Algorithm 3 operates with two distinct implementations) and the smallest one is used to calculate the step-size and to update the Lagrange multipliers. This process is repeated until it reaches a stopping condition. When some Dijkstra's routine does not find a path in Algorithms 3 and 4, the attempt

to get a feasible CSPTP solution fails and the incumbent upper bound remains the same from previous iterations.

Concerning the non-deterministic variants of the repair Algorithms 3 and 4, basically we randomly select the order in which the Dijkstra's algorithm is invoked to obtain each path in line 6 of Algorithm 3 and to select repeated arcs in line 7 of Algorithm 4. The non-deterministic Lagrangian heuristic (NDLH) operates as DLH, but considering non-deterministic calls to Algorithms 3 and 4.

Both DLH and NDLH are embedded into a single Lagrangian framework, i.e., two subgradient algorithms are executed separately and independently, each one implementing distinct heuristics in order to improve incumbent solution values.

3.4 Computational experiments

Models (DN), (FN), and (FN)$^+$, as well DLH and NDLH, were coded using the Java concert technology of CPLEX 12.7. Experiments were performed on a PC Intel Pentium i7, 8 $\times$ 3.60 GHz, 16 GB DDR3 RAM under Linux Ubuntu 18.04. The CPU time limit to execute each instance was set to one hour. For the Lagrangian heuristics, based on practical observations from preliminary numerical experiments, the best set of subgradient parameters are: initial Lagrangian multipliers $\mu^0 = 0$, and starting step-size scaling factor $\lambda_0 = 0.75$, reduced by half at every $\beta = 10$ consecutive iterations without improvement on the lower bound. The `stopCondition()` criteria in Algorithm 2 is either the maximum number of subgradient iterations $K = 500$, or executing up to $3 \times \beta$ iterations with no lower bound improvement, or solution optimality. For this last criterion, as all instances used in the experiments have integer arc costs, we say that a given solution value is optimal if the difference between an upper and a lower bound on the

Algorithm 3: First repair algorithm.

1: **Data:** infeasible solution H of $L(\mu)$, directed graph $D = (V,A)$.
2: **Result:** upper bound UB on the optimal solution value.
3: Let C be a $|A| \times (N-1)$ cost matrix and f_i be the frontier node of T_i in H ($i = 1,\ldots,N$);
4: $UB \leftarrow 0$;
5: `setCost`$(C_{|A|\times(N-1)})$; // initialize the matrix of arc costs
6: **for** ($k \leftarrow 1$ **to** $N-1$) **do**
7: $P \leftarrow$ `dijkstra`$(f_k, f_{k+1}, V, A, C_{|A|\times k})$; // shortest path P from f_k to f_{k+1}
8: $UB \leftarrow UB + \sum_{(i,j)\in P} c_{ij}$; // accumulate sub-path original arc costs
9: $A \leftarrow A - P$; //remove arcs of P from A
10: **end for**

Algorithm 4: Second repair algorithm.

Data: infeasible solution H of $L(\mu)$, directed graph $D = (V,A)$.
Result: upper bound UB on the optimal solution value.
$UB \leftarrow 0$;
for all $(i,j) \in A$ **do**
  $c_{ij} \leftarrow c_{ij} + \mu_{ij}$;
end for
for all $(i,j) \in A$ **do**
  Let m_{ij} be the number of occurrences of arc (i,j) in H;
  if $(m_{ij} = 1)$ **then**
    $UB \leftarrow UB + c_{ij}$;
    Remove (i,j) from A;
  end if
  if $(m_{ij} > 1)$ **then**
    for $k \leftarrow 1$ **to** m_{ij} **do**
      $P \leftarrow \texttt{dijkstra}(i,j,V,A)$; // shortest path P from i to j
      $UB \leftarrow UB + \sum_{(i,j)\in P} c_{ij}$; // accumulate sub-path original arc costs
      $A \leftarrow A - P$; // remove arcs of P from A
    end for
  end if
end for

solution value is less than 1.

We perform comparative analysis considering mathematical programming models (DN), (FN), (FN)$^+$, Lagrangian heuristics DLH and NDLH, state-of-the-art B&B methods (FERONE *et al.*, 2016; FERONE *et al.*, 2019) referred to as BBdf, BBbf, NewBB, and the GRASP algorithm of Ferone *et al.* (2016). All executable files were provided by the authors. In order to obtain a comprehensive feedback from NDLH, we run it 10 times for each problem instance. The number of GRASP iterations is 500, the same of subgradient iterations of our Lagrangian framework.

3.4.1 Benchmark instances

We adopt benchmark data sets of (FERONE *et al.*, 2016). These instances were generated by using an algorithm proposed by (FESTA; PALLOTTINO, 2003), comprising different digraph topologies as complete, sparse, and square/elongated grid digraphs. In the next tables, we group numerical results according to the topology and dimensions of the instances. In each table, we inform the instance identifier *inst*, number of nodes $|V|$, number of arcs $|A|$, number of node disjoint subsets N, and the optimal solution value opt. With respect to individual

results of exact models, we report the CPLEX number of solved subproblems *bb*, the CPLEX number of dual simplex iterations $iter$, and the *CPU* execution time in seconds. For B&B methods (FERONE *et al.*, 2016; FERONE *et al.*, 2019), we give only *CPU* times. As DLH and NDLH achieved the optimal solution for all benchmark data sets, we give only their *CPU* times. For NDLH, we report results for 10 runs and give the worst, average, and best execution times, referred to as CPU^+, $\overline{CPU}$, and CPU^-, respectively. Execution times associated with optimal solutions are in bold font to be distinguished from the ones related to out of memory (OM) error or time limit (TL) exceeded. For the GRASP procedure, we report the incumbent solution value UB and the *CPU* time reached after running 500 iterations. Average column values in the last line of the tables do not consider instances presenting OM error.

For 90 benchmark complete digraphs with 200 nodes (Table 7), NewBB outperformed all exact solution approaches with a very fast average execution time of 0.04 seconds. According to (FERONE *et al.*, 2019), this is due to the fact that their solution approach explores the problem structure of complete digraphs. BBbf and BBdf were not able to solve 40 instances. Concerning IP models, (DN) was the most efficient approach, solving this data set on average execution time of 71.65 seconds. Model (FN) did not solve 3 instances to optimality. On the other hand, $(FN)^+$ spent 347.63 seconds, on average, to solve all these instances. Considering the best execution times observed by the Lagrangian heuristics NDLH and DLH, the average CPU time spent by the Lagrangian framework was of 93.91 seconds, smaller than those of models (FN) and $(FN)^+$. The GRASP algorithm found the optimal solution for only 38 instances.

Concerning 62 benchmark complete digraphs with $|V|$ varying from 250 to 260 nodes (Table 8), the overall performance is similar to the one of Table 7, with NewBB presenting smaller CPU times. The contribution of valid inequalities can be observed when comparing the average execution time of $(FN)^+$ (124.53 seconds) with the one of (FN) (326.06 seconds). However, the best IP approach was again (DN), whose average CPU time was of 66.65 seconds. The best CPU time spent by the Lagrangian heuristic was of 31.62 seconds, on average, which is better than those of IP models. The GRASP algorithm solved to optimality all these instances within an average execution time of 98.68 seconds.

Considering 10 benchmark complete digraphs with 300 (Table 9) and 350 (Table 10) nodes, NewBB presents better CPU times than the other approaches, with average execution time of 0.05 and 0.08 seconds, respectively. On what concerns IP models, (DN) (resp. $((FN)^+$) is the most outstanding approach for digraphs with 300 (resp. 350) nodes. The impact of the proposed

valid inequalities can be accessed when looking at *CPU* columns of (FN) and (FN)$^+$. The Lagrangian framework achieved the optimal solutions faster than any other IP technique, which attests its efficiency and effectiveness. The GRASP algorithm achieved the optimal solution for these 20 digraphs, spending, on average, 142.65 seconds (for instances in Table 9) and 317.53 seconds (for instances in Table 10).

Numerical results for 50 benchmark random sparse digraphs with $|V|$ varying from 100 to 180 nodes (Table 11) underline a slightly competitiveness between solution methods, except BBbf and BBdf. This is not surprising because B&B techniques tend to require a high number of branching operations on these kind of digraphs. The best average CPU times observed were for IP models followed by the Lagrangian framework, which reveals its robustness to achieve optimal solutions in a reasonable CPU time. On the other hand, the GRASP algorithm found the optimal solution for 30 instances.

Concerning benchmark data sets related to 59 grid digraphs (Tables 12–17), NewBB showed difficulty in finding the optimal solution for these instances, which contrasts with the case of complete digraphs. When tackling both random sparse and grid digraphs, NewBB experimented memory limitations. For grids, IP models need at most one second of CPU time each, on average, to achieve optimal solutions. As NewBB, state-of-the-art algorithms BBbf, BBdf, and GRASP failed to solve several instances to optimality. The Lagrangian framework seems to be a promising approach for grids since all optimal solutions are provided in small average execution times, which varies from 1.14 to 16.67 seconds considering the best CPU times observed for NDLH and DLH variants.

3.4.2 New instances

The benchmark data sets addressed all share a common characteristic since they do not require a great computational effort of IP models in the sense that their optimal solutions are found in the root node of the CPLEX branch-and-cut search tree, excepting two instances in Table 17 that require some *bb* nodes to reach their optimal solutions. This motivated us to introduce a new set of instances with 100 and 150 nodes. They were randomly generated according to (ANDRADE; SARAIVA, 2017). For a distinct pair of nodes $i, j \in V$, an arc (i, j) is considered to belong to A according to a given probability $\rho = \frac{2}{3}$. The arc cost c_{ij} is uniformly drawn at random from $[0, 100]$. The number of nodes S to be visited is uniformly drawn at random from $[\frac{|V|}{2}, |V|]$ and the number of subsets N is uniformly drawn at random from $[\frac{S}{4}, \frac{S}{2}]$.

Finally, we set $s \in T_1$ and $t \in T_N$. The remaining nodes to be visited are randomly assigned to $T_2, \ldots, T_{N-1}$, each one having at least one node.

For random digraphs with 100 nodes (Table 18), model $(FN)^+$ spent 53.15 seconds (on average) to solve these instances, whereas (DN) and (FN) needed 66.92 and 170.56 seconds (on average), respectively. It is clear that the proposed valid inequalities improve (FN). Despite the average number of relaxed subproblems solved by $(FN)^+$ be close to the one of (FN), the average number of dual simplex iterations of the former model is approximately one third of that one found by the latter model. NDLH (resp. DLH) found 19 (resp. 4) optimal solutions. State-of-the-art methods BBbf, BBdf, and NewBB failed to solve all these instances due to memory limitation, while GRASP only achieved feasible solutions for them.

Regarding random digraphs with 150 nodes (Table 19), $(FN)^+$ showed to be the most efficient approach while solving them in 452.33 seconds (on average), whereas (DN) and (FN) needed 537.97 and 986.82 seconds, respectively. NDLH achieved the optimal solution for 13 instances, while DLH just for one. Columns related to BBbf, BBdf, and NewBB were omitted because they failed to solve all instances due to memory limitation. The GRASP algorithm did not find any optimal solution.

The numerical experiments carried out on the new data sets clearly indicate that the best branch-and-bound approach from the literature, referred to as NewBB, worked better for benchmark complete digraphs. Nevertheless, it failed to solve some benchmark grid digraphs as well as all new instances. Concerning IP models, (DN) solved all benchmark instances with an average execution time of 55.01 seconds, while $(FN)^+$ in 152.38 seconds. Conversely, for new data sets, $(FN)^+$ solved all instances in 252.74 seconds, while (FN) and (DN) in 578.69 and 302.45 seconds, respectively, showing the efficiency of valid inequalities. Taking into account all 321 instances, (DN) was the best IP model. The Lagrangian framework seems to be a very efficient approach since, for benchmark instances, it found the optimal solution in an average CPU time of 44 seconds (considering the smallest time observed by the non-deterministic version), which is better than the time observed by model (DN) reported above. For new data sets, the Lagrangian framework found the optimal solution for 19 (from a total of 20) digraphs with 100 nodes. For 20 digraphs with 150 nodes, the optimal solution was achieved for 13 instances, which shows how solid the framework is in finding near optimal solutions. The GRASP algorithm from the literature did not find the optimal solution for several instances, being more suitable for benchmark complete digraphs.

3.5 Conclusion

This chapter addresses the NP-Hard constrained shortest path tour problem (CSPTP). We evaluate two models for the problem and introduce valid inequalities allowing to enhance the CPU time performance of the frontier node-based model (FN). By exploring the matrix structure associated with a CSPTP relaxation of (FN), we also develop a Lagrangian-based heuristic framework. We report an exhaustive set of computational experiments on benchmark and new instances. The NewBB approach worked better for benchmark complete digraphs. Nevertheless, it fails to solve the new instances as well as BBbf and BBdf. The Lagrangian framework seems to be very efficient in solving tested instances to optimality in a reasonable CPU time, except for some randomly generated digraphs. Taking into account the whole set of 321 instances, the dummy node-based model (DN) presents the smallest average CPU time.

Table 7 – Results for complete digraphs with $|V| = 200$ (FERONE *et al.*, 2016).

Data set			(DN)		(FN)		$(FN)^+$		BBbf	BBdf	NewBB	NDLH			DLH	GRASP	
inst	*N*	*opt*	*iter*	*CPU*	*iter*	*CPU*	*iter*	*CPU*	*CPU*	*CPU*	*CPU*	CPU^+	$\overline{CPU}$	CPU^-	*CPU*	*UB*	*CPU*
N20-00	40	686	610	**20.27**	7401	**58.87**	607	**15.54**	**5.98**	**6.03**	**0.02**	**11.62**	**9.53**	**7.71**	**8.49**	690	138.57
N20-01	40	673	577	**16.07**	7373	**58.72**	647	**15.69**	**5.97**	**6.02**	**0.02**	**9.09**	**7.57**	**5.94**	**6.55**	684	133.00
N20-02	40	675	668	**17.21**	7395	**64.07**	761	**15.53**	**5.97**	**6.04**	**0.02**	**9.11**	**7.76**	**5.97**	**6.55**	679	130.02
N20-03	40	670	677	**19.65**	7391	**63.09**	689	**15.56**	**5.98**	**6.04**	**0.02**	**10.53**	**9.10**	**7.28**	**8.18**	672	131.66
N20-04	40	707	898	**16.98**	7411	**58.99**	754	**15.60**	**5.98**	**6.02**	**0.02**	**9.28**	**7.77**	**6.09**	**6.84**	711	122.23
N20-05	40	675	605	**16.52**	7283	**57.74**	659	**13.90**	**5.97**	**6.03**	**0.02**	**9.19**	**7.80**	**5.98**	**6.78**	680	132.90
N20-06	40	692	746	**19.37**	7418	**60.96**	716	**15.56**	**6.02**	**6.03**	**0.02**	**12.01**	**10.55**	**8.77**	**9.85**	699	138.64
N20-07	40	679	661	**18.68**	7434	**61.33**	670	**13.03**	**6.02**	**6.04**	**0.02**	**9.03**	**7.64**	**5.73**	**6.51**	681	135.02
N20-08	40	674	684	**19.90**	7253	**59.38**	666	**15.61**	**6.03**	**6.03**	**0.02**	**10.57**	**9.31**	**7.24**	**8.26**	683	132.95
N20-09	40	686	827	**19.21**	7423	**61.53**	681	**15.56**	**6.02**	**6.03**	**0.02**	**9.38**	**7.99**	**5.98**	**6.79**	692	148.04
N30-00	60	1111	781	**49.50**	11392	**146.39**	865	**54.13**	**12.16**	**12.11**	**0.02**	**40.69**	**35.64**	**27.78**	**38.86**	1128	128.75
N30-01	60	1149	1181	**49.28**	11389	**144.37**	1080	**54.92**	**9.69**	**9.67**	**0.02**	**16.26**	**13.26**	**9.47**	**11.06**	1149	**169.09**
N30-02	60	1059	800	**49.58**	11254	**144.49**	824	**56.04**	**9.61**	**9.70**	**0.02**	**16.22**	**13.10**	**9.55**	**10.97**	1064	141.30
N30-03	60	1166	985	**47.76**	11411	**150.69**	1020	**55.99**	**9.60**	**9.70**	**0.02**	**21.32**	**18.40**	**13.35**	**17.18**	1166	**131.08**
N30-04	60	1145	886	**49.12**	11376	**156.27**	861	**57.13**	**9.60**	**9.70**	**0.02**	**19.82**	**17.29**	**13.79**	**18.06**	1146	127.23
N30-05	60	1104	867	**49.42**	11098	**140.44**	882	**63.25**	**12.04**	**12.15**	**0.02**	**293.25**	**247.76**	**186.66**	**230.29**	1106	130.21
N30-06	60	1151	1056	**49.43**	11357	**144.91**	994	**59.12**	**9.62**	**9.72**	**0.02**	**17.17**	**14.28**	**9.61**	**11.52**	1151	**128.81**
N30-07	60	1145	980	**48.84**	11394	**146.14**	1033	**59.67**	**9.66**	**9.78**	**0.02**	**21.17**	**18.07**	**13.50**	**16.52**	1148	132.72
N30-08	60	1175	1188	**50.56**	11418	**146.61**	1190	**60.82**	**9.68**	**9.74**	**0.02**	**21.80**	**18.31**	**12.69**	**16.66**	1180	128.95
N30-09	60	1147	957	**49.09**	11433	**148.04**	1032	**60.92**	**12.09**	**12.28**	**0.02**	**42.26**	**36.32**	**25.61**	**31.19**	1147	**136.56**
N40-00	80	1659	1304	**56.98**	15413	**252.11**	1271	**86.90**	**19.07**	**19.49**	**0.02**	**25.20**	**19.39**	**12.50**	**15.39**	1659	**128.30**
N40-01	80	1625	1218	**58.30**	15366	**239.83**	1172	**83.53**	**15.58**	**16.13**	**0.02**	**32.49**	**26.58**	**18.06**	**22.00**	1625	**146.75**
N40-02	80	1567	1158	**56.61**	15530	**248.51**	1224	**82.72**	**18.99**	**19.65**	**0.02**	**63.63**	**53.24**	**31.88**	**42.82**	1567	**133.66**
N40-03	80	1586	1072	**59.71**	15399	**242.86**	1068	**84.81**	**25.81**	**26.79**	**0.02**	**195.83**	**112.06**	**36.66**	**58.20**	1587	127.09
N40-04	80	1548	1077	**55.65**	15414	**256.47**	1057	**80.13**	**15.60**	**16.51**	**0.02**	**32.66**	**27.28**	**18.07**	**21.39**	1548	**119.81**
N40-05	80	1683	1346	**55.32**	15408	**236.84**	1291	**81.08**	**15.48**	**16.65**	**0.02**	**30.82**	**24.84**	**16.46**	**21.46**	1687	121.73
N40-06	80	1667	1341	**57.24**	15347	**246.58**	1419	**84.15**	**15.52**	**16.81**	**0.02**	**30.82**	**24.84**	**16.44**	**18.80**	1676	129.12
N40-07	80	1648	1230	**56.88**	15414	**256.51**	1267	**87.07**	**15.54**	**16.89**	**0.02**	**31.18**	**27.20**	**18.73**	**21.26**	1648	**140.74**
N40-08	80	1671	1361	**56.28**	15421	**245.72**	1330	**81.69**	**15.56**	**16.95**	**0.02**	**30.76**	**25.46**	**17.50**	**20.46**	1671	**148.51**
N40-09	80	1567	1052	**60.02**	15272	**242.71**	1082	**82.41**	**18.92**	**20.56**	**0.02**	**137.39**	**81.45**	**39.42**	**345.32**	1567	**140.63**
N50-00	100	2103	1425	**69.31**	88037	**171.34**	1439	**111.18**	**41.35**	**44.52**	**0.02**	**138.53**	**108.25**	**73.14**	**94.04**	2103	**111.42**
N50-01	100	2172	1628	**70.26**	81996	**118.19**	1605	**110.60**	**32.57**	**40.15**	**0.02**	**137.36**	**99.70**	**46.88**	**49.37**	2173	131.53
N50-02	100	2119	1556	**69.04**	88830	**153.59**	1566	**111.47**	**32.91**	**40.81**	**0.02**	**141.86**	**121.32**	**87.32**	**146.78**	2121	123.56
N50-03	100	2183	1641	**69.34**	91222	**162.80**	1613	**106.42**	**33.28**	**45.30**	**0.02**	**107.65**	**84.29**	**70.95**	**80.60**	2185	121.04
N50-04	100	2146	1507	**67.60**	91771	**138.30**	1467	**111.67**	**25.98**	**26.32**	**0.02**	**144.20**	**124.03**	**79.05**	**123.20**	2146	**107.77**
N50-05	100	2251	1787	**70.02**	83653	**138.49**	1735	**106.25**	**30.74**	**31.07**	**0.02**	**84.66**	**69.19**	**44.96**	**55.45**	2251	**130.66**
N50-06	100	2247	1646	**73.26**	89066	**136.27**	1697	**110.33**	**43.03**	**44.94**	**0.02**	**213.28**	**129.27**	**77.04**	**90.24**	2247	**135.46**
N50-07	100	2177	1596	**69.75**	87898	**153.43**	1634	**118.64**	**29.78**	**31.46**	**0.02**	**91.36**	**74.09**	**45.67**	**56.81**	2178	120.98
N50-08	100	2154	1620	**68.21**	82858	**127.19**	1606	**105.18**	**25.63**	**26.74**	**0.02**	**49.00**	**40.31**	**29.53**	**31.19**	2155	123.34
N50-09	100	2134	1457	**69.49**	87021	**141.55**	1467	**110.06**	**39.72**	**41.23**	**0.02**	**95.50**	**78.56**	**46.00**	**56.30**	2135	118.09
N60-00	120	2703	2030	**44.73**	106418	**152.58**	1979	**53.28**	**77.31**	**83.79**	**0.03**	**3513.99**	**2267.35**	**446.65**	**115.28**	2703	**127.46**
N60-01	120	2678	1784	**43.84**	118370	**235.34**	1742	**52.08**	**36.50**	**42.68**	**0.02**	**59.23**	**46.39**	**23.29**	**26.30**	2678	**127.86**
N60-02	120	2700	1983	**45.16**	112570	**241.96**	1945	**55.73**	**85.59**	**100.84**	**0.02**	**251.43**	**188.91**	**84.82**	**86.36**	2700	**124.86**
N60-03	120	2671	1858	**44.53**	121066	**274.47**	1834	**51.23**	**61.62**	**66.80**	**0.02**	**187.99**	**153.48**	**95.10**	**113.32**	2672	126.77
N60-04	120	2636	1606	**43.89**	128931	**319.91**	1532	**55.54**	**38.00**	**37.19**	**0.02**	**52.93**	**42.77**	**30.11**	**34.90**	2636	**121.67**
N60-05	120	2728	1930	**43.81**	110428	**168.81**	1878	**52.61**	**39.78**	**44.50**	**0.02**	**259.79**	**183.47**	**52.46**	**97.85**	2730	119.51
N60-06	120	2764	1934	**45.80**	108060	**207.01**	1913	**55.73**	**45.77**	**57.93**	**0.02**	**271.28**	**210.65**	**125.25**	**154.96**	2764	**124.56**
N60-07	120	2711	1981	**45.28**	119543	**251.09**	1950	**53.19**	**80.54**	**188.60**	**0.03**	**290.79**	**189.26**	**81.79**	**144.18**	2712	128.52
N60-08	120	2708	1892	**44.44**	118456	**229.99**	1856	**53.15**	**191.43**	**187.52**	**0.04**	**1116.20**	**480.13**	**188.64**	**1891.71**	2708	**122.89**
N60-09	120	2723	1761	**44.48**	107134	**185.58**	1753	**52.68**	**49.71**	**51.18**	**0.02**	**189.05**	**146.89**	**92.52**	**97.03**	2723	**125.62**
N70-00	140	3219	2004	**50.47**	155414	**419.18**	1985	**59.58**	OM	OM	**0.02**	**311.55**	**244.38**	**95.82**	**99.68**	3219	**115.82**
N70-01	140	3269	2158	**51.52**	140017	**387.74**	2086	**62.67**	OM	OM	**0.02**	**306.99**	**247.46**	**64.43**	**219.05**	3269	**114.17**
N70-02	140	3267	2179	**51.27**	159659	**503.06**	2166	**61.91**	OM	OM	**0.02**	**76.47**	**59.58**	**40.32**	**44.66**	3267	**111.19**
N70-03	140	3300	2284	**50.05**	166303	**567.22**	2264	**64.04**	OM	OM	**0.03**	**371.35**	**250.26**	**98.57**	**98.90**	3300	**118.25**
N70-04	140	3328	2357	**52.03**	137369	**335.40**	2354	**63.84**	OM	OM	**0.02**	**244.32**	**182.65**	**94.01**	**100.76**	3328	**105.88**
N70-05	140	3177	2015	**51.76**	138517	**361.07**	2004	**63.47**	OM	OM	**0.02**	**68.43**	**52.42**	**28.29**	**31.62**	3177	**124.87**
N70-06	140	3385	2345	**49.85**	157306	**482.26**	2313	**60.22**	OM	OM	**0.03**	**471.98**	**320.30**	**100.29**	**146.94**	3387	127.53
N70-07	140	3243	2232	**50.38**	130655	**325.49**	2214	**63.29**	OM	OM	**0.03**	**318.44**	**257.83**	**109.05**	**189.29**	3243	**120.52**
N70-08	140	3245	2108	**52.43**	160719	**501.01**	2061	**64.38**	OM	OM	**0.03**	**229.94**	**183.60**	**96.16**	**99.76**	3247	98.35
N70-09	140	3329	2245	**52.70**	157332	**435.37**	2164	**61.92**	OM	OM	**0.02**	**229.62**	**174.49**	**64.80**	**74.67**	3329	**116.50**
N80-00	160	3855	2579	**57.08**	239122	**1122.59**	233350	**388.87**	OM	OM	**0.02**	**382.54**	**250.14**	**134.79**	**170.31**	3855	**114.29**
N80-01	160	3828	2402	**58.58**	174991	**540.49**	205321	**472.42**	OM	OM	**0.02**	**474.32**	**345.52**	**124.83**	**246.31**	3828	**109.34**
N80-02	160	3999	2901	**57.03**	190444	**638.49**	234701	**554.54**	OM	OM	**0.04**	**577.57**	**420.17**	**228.79**	**237.51**	4009	123.35
N80-03	160	3917	2694	**57.65**	194128	**717.82**	211513	**434.09**	OM	OM	**0.02**	**394.64**	**317.12**	**178.56**	**232.81**	3917	**135.06**
N80-04	160	3857	2527	**58.74**	227172	**982.86**	214026	**476.12**	OM	OM	**0.07**	**378.71**	**303.85**	**141.47**	**172.12**	3866	109.43
N80-05	160	3811	2438	**56.77**	204875	**831.21**	175390	**345.43**	OM	OM	**0.03**	**495.52**	**340.00**	**204.11**	**226.46**	3811	**106.01**
N80-06	160	3931	2719	**59.37**	188922	**672.89**	215498	**460.85**	OM	OM	**0.31**	**480.46**	**374.00**	**183.93**	**198.93**	3935	118.67
N80-07	160	3799	2469	**54.85**	222233	**915.84**	210932	**495.67**	OM	OM	**0.07**	**477.44**	**337.48**	**197.93**	**276.78**	3814	99.46
N80-08	160	3805	2516	**57.25**	216598	**909.07**	196714	**397.36**	OM	OM	**0.02**	**281.62**	**237.17**	**77.96**	**97.56**	3805	**113.92**
N80-09	160	3794	2482	**57.44**	195626	**763.32**	174341	**345.60**	OM	OM	**0.02**	**459.20**	**332.82**	**123.05**	**167.59**	3794	**118.11**
N90-00	180	4390	2820	**129.75**	360190	**2215.67**	290262	**903.43**	OM	OM	**0.03**	**327.28**	**240.02**	**107.21**	**173.18**	4391	84.18
N90-01	180	4415	2754	**125.04**	315993	**1704.58**	238648	**731.90**	OM	OM	**0.03**	**579.39**	**373.86**	**138.65**	**184.59**	4417	84.85
N90-02	180	4461	2948	**129.25**	318702	**1919.90**	289499	**888.71**	OM	OM	**0.04**	**435.54**	**311.27**	**156.17**	**189.66**	4463	104.43
N90-03	180	4518	3018	**133.27**	261149	**1381.65**	271300	**858.62**	OM	OM	**0.03**	**2104.07**	**1198.00**	**203.84**	**385.53**	4518	**110.57**
N90-04	180	4338	2593	**123.33**	290197	**1629.73**	227608	**661.32**	OM	OM	**0.03**	**562.85**	**345.17**	**150.67**	**199.91**	4348	101.77
N90-05	180	4349	2695	**126.28**	323524	**1871.28**	350963	**1167.85**	OM	OM	**0.03**	**454.24**	**305.18**	**139.36**	**172.86**	4354	102.50
N90-06	180	4498	2896	**130.50**	283329	**1543.06**	312975	**843.94**	OM	OM	**0.03**	**420.24**	**289.99**	**137.31**	**140.02**	4509	118.93
N90-07	180	4342	2806	**127.67**	348401	**1879.16**	198908	**521.32**	OM	OM	**0.56**	**1461.57**	**942.27**	**465.08**	**388.00**	4346	110.28
N90-08	180	4360	2731	**125.57**	244773	**1219.09**	304922	**980.69**	OM	OM	**0.02**	**432.01**	**323.94**	**139.62**	**239.73**	4360	**112.80**
N90-09	180	4425	2790	**129.59**	341828	**2027.50**	320304	**904.79**	OM	OM	**0.04**	**672.47**	**408.51**	**138.94**	**157.24**	4430	99.34
N100-00	200	5004	3291	**165.19**	TL	TL	436822	**1250.84**	OM	OM	**0.04**	**399.92**	**320.87**	**199.43**	**290.38**	5006	48.56
N100-01	200	4963	3013	**168.91**	440563	**3088.31**	351482	**1141.14**	OM	OM	**0.04**	**530.04**	**390.67**	**124.58**	**251.72**	4971	45.72
N100-02	200	5086	3483	**167.27**	424128	**2471.39**	394851	**1176.66**	OM	OM	**0.10**	**993.02**	**663.53**	**304.87**	**440.21**	5094	51.68
N100-03	200	5095	3394	**168.58**	468318	**3260.55**	366850	**1216.55**	OM	OM	**0.07**	**1329.73**	**701.93**	**206.95**	**430.79**	5098	50.40
N100-04	200	4964	3060	**174.89**	472001	**3370.92**	460888	**1343.79**	OM	OM	**0.03**	**763.86**	**488.14**	**227.71**	**254.83**	4973	45.30
N100-05	200	4953	3035	**170.46**	402343	**2649.34**	433976	**1614.06**	OM	OM	**0.03**	**522.24**	**384.20**	**176.10**	**197.21**	4958	45.99
N100-06	200	5100	3398	**164.64**	442132	**2931.00**	347014	**1157.09**	OM	OM	**0.21**	**796.18**	**493.49**	**207.37**	**338.02**	5122	51.20
N100-07	200	5028	3343	**167.07**	TL	TL	682240	**3315.77**	OM	OM	**0.75**	**1087.77**	**687.75**	**306.44**	**434.59**	5041	50.07
N100-08	200	4994	3180	**173.23**	415768	**2860.34**	363244	**1150.11**	OM	OM	**0.05**	**1262.11**	**559.66**	**213.26**	**346.58**	4997	50.23
N100-09	200	4991	3240	**168.82**	TL	TL	371429	**1256.75**	OM	OM	**0.20**	**2204.07**	**945.49**	**294.26**	**435.81**	4996	50.87
Average			1268.3	47.83	47292.3	164.98	1260.5	64.10	27.62	31.70	0.02	172.33	115.56	48.71	92.20	–	130.15

Table 8 – Results for complete digraphs with $|V| \in \{250, 252, 254, 256, 258, 260\}$ (FERONE *et al.*, 2016).

Data set			(DN)		(FN)		$(FN)^+$		BBbf	BBdf	NewBB	NDLH			DLH	GRASP
inst	*N*	*opt*	*iter*	*CPU*	*iter*	*CPU*	*iter*	*CPU*	*CPU*	*CPU*	*CPU*	CPU^+	$\overline{CPU}$	CPU^-	*CPU*	*CPU*
250N25-00	62	1350	1024	**76.88**	14876	**300.92**	1014	**111.00**	**17.54**	**17.42**	**0.03**	**27.49**	**23.76**	**16.57**	**16.64**	**103.45**
250N25-01	62	1330	1018	**75.78**	14828	**304.80**	1011	**110.48**	**22.00**	**21.58**	**0.04**	**75.84**	**62.33**	**45.51**	**43.93**	**86.68**
250N25-02	62	1367	1101	**72.52**	14863	**300.52**	1098	**106.56**	**17.61**	**17.33**	**0.04**	**30.12**	**25.37**	**18.26**	**17.51**	**106.01**
250N25-03	62	1394	1321	**74.66**	14998	**303.44**	1264	**111.10**	**21.80**	**21.70**	**0.04**	**85.44**	**69.39**	**50.75**	**75.67**	**112.55**
250N25-04	62	1355	992	**74.74**	14815	**307.06**	1026	**108.85**	**17.46**	**17.25**	**0.03**	**31.50**	**24.25**	**16.93**	**17.85**	**86.71**
250N25-05	62	1336	1046	**75.90**	14893	**315.93**	1004	**113.42**	**17.49**	**17.39**	**0.03**	**29.35**	**24.47**	**16.99**	**17.79**	**87.16**
250N25-07	62	1405	1244	**75.37**	14897	**296.18**	1244	**110.61**	**17.45**	**17.41**	**0.03**	**34.78**	**29.43**	**20.40**	**22.20**	**95.43**
250N25-08	62	1368	1121	**77.08**	14896	**308.63**	1077	**116.52**	**17.46**	**17.44**	**0.03**	**36.65**	**28.67**	**20.57**	**22.94**	**96.09**
250N25-09	62	1325	1112	**76.66**	15023	**310.36**	1112	**121.89**	**21.73**	**21.74**	**0.04**	**34.78**	**30.37**	**21.67**	**24.12**	**84.54**
250N25-10	62	61	252	**74.95**	14907	**298.41**	212	**113.05**	**17.34**	**17.33**	**0.03**	**33.51**	**26.13**	**16.51**	**18.44**	**21.61**
250N25-11	62	61	219	**73.83**	14907	**320.56**	228	**113.06**	**17.35**	**17.29**	**0.03**	**36.14**	**26.36**	**17.54**	**20.88**	**22.71**
252N25-00	63	1433	1225	**74.14**	15301	**319.44**	1180	**120.32**	**17.95**	**17.95**	**0.04**	**38.64**	**31.56**	**22.83**	**24.25**	**85.06**
252N25-01	63	1387	1061	**75.23**	15118	**298.63**	1068	**117.66**	**17.87**	**17.94**	**0.04**	**39.02**	**31.50**	**24.33**	**25.50**	**74.52**
252N25-02	63	1362	1032	**75.36**	15125	**305.02**	1027	**117.92**	**17.84**	**17.92**	**0.04**	**36.78**	**31.29**	**22.56**	**25.61**	**95.31**
252N25-03	63	1380	1067	**77.75**	15234	**330.91**	1023	**125.11**	**17.90**	**17.88**	**0.04**	**39.87**	**32.72**	**24.85**	**26.39**	**77.98**
252N25-04	63	1435	1053	**79.54**	15250	**318.93**	1079	**119.61**	**17.89**	**17.88**	**0.04**	**42.14**	**33.43**	**22.25**	**26.45**	**90.00**
252N25-05	63	1383	1010	**78.24**	15109	**314.03**	1024	**120.80**	**22.22**	**22.36**	**0.04**	**77.87**	**62.15**	**44.45**	**54.79**	**81.35**
252N25-06	63	1403	1208	**80.04**	15383	**321.83**	1207	**123.43**	**17.82**	**17.99**	**0.04**	**41.99**	**32.85**	**21.77**	**25.97**	**82.68**
252N25-07	63	1446	1123	**77.68**	15384	**323.89**	1118	**119.84**	**22.27**	**22.36**	**0.04**	**88.70**	**70.41**	**52.00**	**63.57**	**99.61**
252N25-08	63	1393	1172	**77.09**	15441	**309.43**	1171	**120.52**	**26.69**	**26.85**	**0.04**	**86.44**	**73.11**	**54.92**	**85.28**	**98.12**
252N25-09	63	1415	1200	**79.01**	15108	**307.48**	1150	**115.15**	**17.87**	**17.94**	**0.04**	**37.46**	**31.04**	**20.77**	**25.52**	**68.40**
254N25-00	63	1405	1136	**79.47**	15170	**333.14**	1100	**129.20**	**18.09**	**18.36**	**0.04**	**85.88**	**67.05**	**44.05**	**53.50**	**93.94**
254N25-02	63	1347	952	**82.13**	15355	**331.46**	925	**121.60**	**22.85**	**22.83**	**0.04**	**91.14**	**74.25**	**52.28**	**81.00**	**66.37**
254N25-03	63	1324	943	**79.36**	15307	**315.01**	947	**126.02**	**22.78**	**22.86**	**0.04**	**82.93**	**67.67**	**47.20**	**57.04**	**76.36**
254N25-04	63	1387	1063	**80.91**	15267	**316.49**	1018	**123.88**	**22.81**	**22.86**	**0.04**	**42.44**	**35.81**	**24.63**	**28.42**	**82.87**
254N25-05	63	1389	1150	**79.16**	15428	**316.29**	1153	**125.90**	**22.78**	**22.84**	**0.04**	**92.34**	**71.62**	**46.03**	**63.32**	**98.07**
254N25-06	63	1394	1096	**79.31**	15217	**313.05**	1114	**125.84**	**18.31**	**18.39**	**0.04**	**43.09**	**34.40**	**23.45**	**28.65**	**92.97**
254N25-07	63	1401	1146	**80.47**	15234	**315.95**	1130	**125.04**	**18.35**	**18.48**	**0.04**	**57.95**	**42.40**	**27.77**	**32.78**	**108.51**
254N25-08	63	1371	1041	**77.84**	15445	**319.19**	1028	**121.40**	**18.27**	**18.80**	**0.04**	**33.15**	**27.20**	**18.32**	**20.14**	**137.08**
254N25-09	63	1423	1190	**81.86**	15228	**324.23**	1172	**128.48**	**18.30**	**18.88**	**0.04**	**40.73**	**33.23**	**24.60**	**27.66**	**118.48**
254N25-11	63	62	204	**79.85**	15294	**315.05**	207	**118.47**	**18.03**	**18.79**	**0.03**	**39.52**	**28.21**	**17.44**	**19.42**	**29.64**
254N25-13	63	1423	1159	**79.70**	15377	**317.38**	1162	**125.27**	**22.76**	**23.88**	**0.04**	**36.76**	**28.07**	**18.57**	**21.19**	**117.42**
256N25-00	64	1420	1123	**80.47**	15723	**344.47**	1131	**128.47**	**19.72**	**21.00**	**0.04**	**45.90**	**35.22**	**24.38**	**27.89**	**117.62**
256N25-01	64	1428	1163	**87.43**	15800	**361.92**	1152	**124.41**	**19.78**	**20.97**	**0.04**	**45.86**	**36.14**	**26.14**	**30.60**	**114.90**
256N25-02	64	1404	1108	**82.93**	15849	**353.86**	1126	**126.56**	**24.48**	**25.94**	**0.04**	**219.77**	**102.70**	**59.29**	**463.23**	**86.89**
256N25-03	64	1441	1140	**77.61**	15769	**350.29**	1160	**127.70**	**20.78**	**21.36**	**0.04**	**46.06**	**35.45**	**25.81**	**27.54**	**127.72**
256N25-04	64	1376	938	**86.53**	15767	**337.94**	935	**123.47**	**21.00**	**21.56**	**0.04**	**39.00**	**29.52**	**20.27**	**23.55**	**90.44**
256N25-05	64	1397	1040	**76.03**	15788	**343.48**	1051	**122.61**	**25.88**	**31.49**	**0.04**	**42.42**	**32.34**	**22.42**	**23.97**	**97.05**
256N25-06	64	1401	1149	**78.84**	15774	**344.73**	1140	**134.56**	**21.22**	**23.57**	**0.04**	**45.72**	**35.96**	**25.75**	**30.02**	**83.00**
256N25-07	64	1426	1054	**87.87**	15775	**342.98**	1076	**127.77**	**21.29**	**23.23**	**0.04**	**65.78**	**48.18**	**29.63**	**31.25**	**92.38**
256N25-08	64	1318	883	**79.18**	15810	**335.16**	879	**130.81**	**24.80**	**28.18**	**0.04**	**116.06**	**82.12**	**55.05**	**498.92**	**108.02**
256N25-09	64	1419	1156	**79.15**	15792	**332.19**	1191	**124.61**	**20.11**	**23.06**	**0.04**	**43.67**	**35.10**	**25.34**	**28.78**	**134.05**
258N25-00	64	1373	994	**40.84**	15828	**342.05**	1009	**133.12**	**20.76**	**23.33**	**0.04**	**44.29**	**36.30**	**26.82**	**29.57**	**146.49**
258N25-01	64	1454	1165	**39.21**	15912	**337.35**	1171	**131.54**	**21.74**	**21.09**	**0.04**	**45.16**	**36.27**	**25.47**	**28.90**	**97.89**
258N25-02	64	1380	963	**41.53**	15936	**351.68**	953	**133.06**	**25.69**	**26.13**	**0.04**	**88.34**	**69.68**	**46.49**	**54.68**	**128.55**
258N25-03	64	1393	1127	**40.15**	15921	**339.60**	1114	**131.34**	**21.24**	**21.48**	**0.04**	**48.20**	**37.65**	**27.36**	**30.30**	**101.31**
258N25-04	64	1447	1124	**42.50**	15905	**339.93**	1097	**140.70**	**21.27**	**21.52**	**0.04**	**48.45**	**38.80**	**29.10**	**32.29**	**112.51**
258N25-05	64	1391	1037	**40.07**	15897	**347.12**	1033	**132.25**	**21.34**	**22.88**	**0.04**	**39.18**	**30.37**	**20.66**	**22.41**	**117.38**
258N25-06	64	1405	1136	**40.48**	15879	**334.50**	1147	**127.82**	**21.58**	**22.66**	**0.04**	**44.51**	**36.38**	**26.47**	**27.79**	**122.44**
258N25-07	64	1432	1271	**43.61**	15900	**339.02**	1256	**133.85**	**21.65**	**22.87**	**0.04**	**46.56**	**36.75**	**26.67**	**29.13**	**133.67**
258N25-08	64	1398	1070	**41.65**	15915	**344.26**	1060	**132.72**	**21.27**	**22.71**	**0.04**	**40.89**	**30.52**	**20.11**	**22.35**	**118.49**
258N25-09	64	1420	975	**41.01**	15896	**345.69**	945	**132.43**	**21.66**	**22.38**	**0.04**	**38.47**	**30.49**	**20.68**	**21.77**	**118.64**
260N25-00	65	1424	1174	**44.74**	16122	**315.50**	1178	**130.80**	**25.29**	**26.39**	**0.04**	**43.99**	**32.85**	**22.02**	**24.08**	**133.79**
260N25-01	65	1473	1244	**43.54**	16134	**318.26**	1240	**129.72**	**25.21**	**26.31**	**0.04**	**47.18**	**37.76**	**27.03**	**28.97**	**129.43**
260N25-02	65	1399	1047	**40.61**	16084	**319.39**	1050	**138.61**	**26.58**	**26.21**	**0.04**	**49.62**	**39.03**	**27.33**	**28.63**	**86.24**
260N25-03	65	1400	1072	**41.98**	16108	**341.50**	1063	**133.83**	**26.61**	**26.38**	**0.04**	**51.05**	**40.58**	**30.36**	**31.93**	**83.70**
260N25-04	65	1491	1096	**42.99**	16097	**324.81**	1120	**128.89**	**26.41**	**26.43**	**0.04**	**50.24**	**40.63**	**28.68**	**29.80**	**144.88**
260N25-05	65	1433	1133	**41.01**	16348	**337.31**	1121	**130.21**	**31.52**	**31.48**	**0.04**	**100.75**	**79.95**	**60.12**	**61.26**	**94.67**
260N25-06	65	1488	1354	**41.85**	16416	**338.44**	1325	**130.70**	**24.12**	**26.67**	**0.04**	**49.39**	**38.21**	**27.25**	**29.04**	**98.88**
260N25-07	65	1483	1332	**43.08**	16501	**350.54**	1352	**133.21**	**29.19**	**31.15**	**0.04**	**104.91**	**81.42**	**57.80**	**59.67**	**104.93**
260N25-08	65	1463	1250	**41.45**	16321	**339.04**	1234	**130.98**	**39.55**	**41.57**	**0.04**	**698.84**	**464.34**	**121.87**	**424.83**	**102.94**
260N25-09	65	1384	1026	**41.38**	16171	**329.17**	1027	**125.89**	**37.79**	**36.59**	**0.04**	**154.17**	**116.20**	**59.38**	**92.99**	**98.98**
Average			1064.9	66.65	15545.4	326.06	1058.5	124.53	21.85	22.49	0.04	67.34	50.60	31.65	55.01	98.68

Table 9 – Results for complete digraphs with $|V| = 300$ and $N = 75$ (FERONE *et al.*, 2016).

Data set		(DN)		(FN)		$(FN)^+$		BBbf	BBdf	NewBB	NDLH			DLH	GRASP
inst	*opt*	*iter*	*CPU*	*iter*	*CPU*	*iter*	*CPU*	*CPU*	*CPU*	*CPU*	CPU^+	$\overline{CPU}$	CPU^-	*CPU*	*CPU*
N25t40-00	1615	1452	**147.47**	21802	**684.71**	1453	**203.56**	**36.52**	**38.89**	**0.05**	**68.03**	**49.89**	**36.83**	**37.97**	**130.09**
N25t40-01	1598	1345	**159.80**	21523	**688.89**	1354	**236.33**	**36.33**	**36.97**	**0.05**	**82.90**	**63.94**	**44.32**	**45.54**	**134.17**
N25t40-02	1647	1416	**165.85**	21914	**709.95**	1408	**252.40**	**36.46**	**38.04**	**0.05**	**74.22**	**52.76**	**32.9**	**33.04**	**149.90**
N25t40-03	1660	1503	**167.26**	21841	**719.15**	1513	**241.61**	**36.66**	**37.46**	**0.06**	**86.86**	**63.84**	**35.56**	**37.04**	**162.33**
N25t40-04	1571	1257	**174.70**	21608	**688.52**	1255	**255.45**	**36.60**	**37.06**	**0.05**	**86.60**	**63.98**	**37.24**	**36.80**	**147.52**
N25t40-05	1600	1334	**148.85**	21858	**707.73**	1344	**245.32**	**44.60**	**45.40**	**0.05**	**81.27**	**63.88**	**49.25**	**43.73**	**114.56**
N25t40-06	1642	1395	**162.52**	21925	**706.32**	1340	**253.75**	**36.59**	**37.19**	**0.05**	**87.37**	**63.69**	**49.04**	**44.81**	**134.83**
N25t40-07	1639	1410	**153.08**	22018	**716.35**	1409	**244.62**	**36.63**	**37.25**	**0.05**	**82.73**	**63.86**	**41.91**	**44.10**	**165.76**
N25t40-08	1607	1273	**172.09**	21598	**687.33**	1238	**250.16**	**36.53**	**37.35**	**0.05**	**83.83**	**63.67**	**41.59**	**42.81**	**131.79**
N25t40-09	1644	1383	**165.41**	21838	**715.17**	1345	**253.02**	**44.64**	**45.77**	**0.05**	**165.31**	**126.27**	**94.16**	**92.75**	**155.55**
Average		1376.8	161.70	21792.5	702.41	1365.9	243.62	38.16	39.14	0.05	89.91	67.58	46.28	45.86	142.65

Table 10 – Results for complete digraphs with $|V| = 350$ and $N = 87$ (FERONE *et al.*, 2016).

Data set		(DN)		(FN)		$(FN)^+$		BBbf	BBdf	NewBB	NDLH			DLH	GRASP
inst	*opt*	*iter*	*CPU*	*iter*	*CPU*	*iter*	*CPU*	*CPU*	*CPU*	*CPU*	CPU^+	$\overline{CPU}$	CPU^-	*CPU*	*CPU*
N25t40-00	1874	1697	**258.90**	140458	**240.32**	1701	**136.62**	OM	OM	**0.09**	**487.53**	**378.42**	**186.42**	**260.91**	**327.82**
N25t40-01	1810	1527	**251.21**	139066	**238.40**	1459	**131.45**	OM	OM	**0.08**	**136.03**	**98.71**	**67.82**	**69.61**	**305.98**
N25t40-02	1903	1707	**369.03**	128206	**210.07**	1696	**135.67**	OM	OM	**0.08**	**157.20**	**125.95**	**74.32**	**76.08**	**344.90**
N25t40-03	1848	1626	**389.76**	134296	**220.12**	1617	**135.10**	OM	OM	**0.08**	**157.84**	**125.74**	**70.67**	**79.95**	**283.05**
N25t40-04	1895	1613	**384.21**	154890	**403.07**	1591	**137.34**	OM	OM	**0.08**	**137.10**	**103.02**	**70.70**	**71.27**	**280.31**
N25t40-05	1872	1764	**316.17**	139692	**233.99**	1767	**129.72**	OM	OM	**0.08**	**164.68**	**129.00**	**71.68**	**77.94**	**396.47**
N25t40-06	1904	1679	**315.01**	141827	**283.72**	1633	**133.73**	OM	OM	**0.08**	**161.63**	**128.48**	**72.15**	**79.14**	**322.02**
N25t40-07	1870	1677	**304.33**	139389	**207.95**	1600	**131.91**	OM	OM	**0.08**	**159.65**	**125.47**	**92.52**	**99.05**	**320.24**
N25t40-08	1852	1685	**308.92**	137704	**209.58**	1633	**132.65**	OM	OM	**0.08**	**145.31**	**110.48**	**71.60**	**74.08**	**300.30**
N25t40-09	1831	1696	**305.97**	138112	**234.63**	1683	**131.81**	OM	OM	**0.10**	**496.52**	**388.17**	**261.44**	**276.13**	**294.23**
Average		1667.1	320.35	139364.0	248.18	1638.0	133.60	–	–	0.08	220.35	171.34	103.93	116.42	317.53

Table 11 – Results for random digraphs with $|V| = 100, 120, 140, 160$ and 180, whose number of arcs $|A| = 500, 600, 700, 800$ and 900, and whose number of node disjoint subsets $N = 20, 24, 28, 32$ and 36, respectively (FERONE *et al.*, 2016).

Data set		(DN)		(FN)		$(FN)^+$		BBbf	BBdf	NewBB	NDLH			DLH	GRASP	
inst	*opt*	*iter*	*CPU*	*iter*	*CPU*	*iter*	*CPU*	*CPU*	*CPU*	*CPU*	CPU^+	$\overline{CPU}$	CPU^-	*CPU*	*UB*	*CPU*
100N20-00	1981	338	**0.23**	1817	**0.32**	331	**0.25**	**0.04**	**0.15**	**0.01**	**0.54**	**0.22**	**0.14**	**0.17**	1981	**0.86**
100N20-01	1980	406	**0.24**	1850	**0.24**	340	**0.06**	**0.07**	**0.05**	**0.01**	**1.17**	**0.77**	**0.35**	**1.44**	1980	**0.87**
100N20-02	1989	345	**0.21**	2038	**0.29**	340	**0.06**	**0.21**	**0.42**	**0.01**	**0.75**	**0.48**	**0.32**	**0.38**	1989	**0.69**
100N20-03	2007	414	**0.33**	1948	**0.33**	383	**0.14**	**0.51**	**1.28**	**0.01**	**24.03**	**9.42**	**1.23**	**2.33**	2007	**0.91**
100N20-04	1620	263	**0.30**	1921	**0.26**	265	**0.06**	**0.06**	**0.12**	**0.01**	**0.82**	**0.58**	**0.26**	**0.44**	1620	**0.79**
100N20-05	2172	350	**0.23**	1853	**0.31**	375	**0.06**	**0.17**	**0.21**	**0.01**	**0.52**	**0.33**	**0.16**	**0.26**	2172	**0.89**
100N20-06	1965	374	**0.25**	1881	**0.27**	352	**0.06**	**0.27**	**1.14**	**0.01**	**0.77**	**0.45**	**0.27**	**0.33**	1965	**0.92**
100N20-07	1729	306	**0.26**	1785	**0.25**	289	**0.06**	**0.03**	**0.05**	**0.01**	**0.22**	**0.16**	**0.09**	**0.14**	1729	**0.80**
100N20-08	2135	450	**0.27**	1955	**0.30**	398	**0.06**	**0.12**	**0.61**	**0.01**	**0.42**	**0.34**	**0.21**	**0.29**	2156	0.87
100N20-09	2194	457	**0.27**	2087	**0.27**	453	**0.06**	**0.39**	**0.37**	**0.01**	**2.83**	**1.82**	**0.83**	**1.63**	2194	**0.98**
120N20-00	2380	606	**0.50**	3322	**0.50**	566	**0.28**	**2.87**	**4.05**	**0.02**	**2.09**	**1.33**	**0.59**	**1.10**	2380	**1.30**
120N20-01	2719	514	**0.40**	2966	**0.45**	519	**0.12**	**0.10**	**0.96**	**0.01**	**1.16**	**0.64**	**0.33**	**1.04**	2719	**1.31**
120N20-02	2872	542	**0.45**	2740	**0.45**	552	**0.38**	**0.08**	**0.27**	**0.01**	**0.94**	**0.47**	**0.18**	**0.46**	2872	**1.23**
120N20-03	2462	662	**0.48**	3122	**0.52**	601	**0.15**	**0.43**	**0.55**	**0.01**	**37.65**	**25.01**	**9.67**	**11.55**	2462	**1.25**
120N20-04	2445	548	**0.50**	3037	**0.45**	510	**0.29**	**0.21**	**0.40**	**0.01**	**2.31**	**1.48**	**0.72**	**1.59**	2445	**1.29**
120N20-05	2747	611	**0.49**	3069	**0.51**	576	**0.08**	**0.34**	**0.78**	**0.03**	**2.06**	**1.16**	**0.44**	**1.30**	2747	**1.29**
120N20-06	2780	543	**0.34**	2899	**0.45**	548	**0.28**	**0.10**	**0.10**	**0.01**	**2.57**	**1.21**	**0.34**	**1.25**	2780	**1.46**
120N20-07	2767	595	**0.47**	3040	**0.58**	583	**0.35**	**9.91**	**22.95**	**0.26**	**1.51**	**0.93**	**0.63**	**0.75**	2771	1.27
120N20-08	2481	609	**0.45**	3427	**0.55**	571	**0.29**	**3.28**	**8.46**	**0.14**	**2.05**	**1.24**	**0.77**	**1.61**	2491	1.48
120N20-09	2700	622	**0.52**	3110	**0.49**	592	**0.40**	**2.40**	**7.34**	**0.07**	**8.41**	**6.60**	**4.73**	**15.49**	2700	**1.33**
140N20-00	3621	943	**0.73**	4681	**0.93**	865	**0.50**	**151.32**	**305.20**	**2.34**	**12.70**	**7.79**	**4.25**	**5.17**	3634	1.84
140N20-01	3212	648	**0.67**	3816	**0.67**	588	**0.49**	**0.28**	**1.03**	**0.08**	**1.51**	**1.13**	**0.55**	**0.66**	3215	1.76
140N20-02	2956	729	**0.77**	4332	**1.09**	759	**0.53**	**0.45**	**1.93**	**0.01**	**4.29**	**2.40**	**1.24**	**2.63**	2956	**1.85**
140N20-03	3254	653	**0.66**	4169	**0.88**	608	**0.49**	**1.26**	**5.34**	**0.02**	**2.91**	**1.96**	**0.78**	**1.53**	3254	**1.75**
140N20-04	3152	835	**0.63**	4825	**1.06**	888	**0.46**	**21.34**	**568.24**	**0.39**	**3.02**	**1.69**	**0.81**	**1.20**	3192	2.00
140N20-05	3000	803	**0.74**	4345	**1.03**	779	**0.58**	**21.02**	**92.57**	**0.05**	**56.55**	**43.70**	**8.61**	**44.45**	3000	**2.05**
140N20-06	3190	777	**0.76**	4466	**0.89**	775	**0.49**	**5.23**	**45.04**	**0.07**	**3.04**	**2.30**	**1.48**	**1.16**	3207	1.80
140N20-07	2774	596	**0.66**	3830	**0.79**	550	**0.15**	**0.35**	**0.80**	**0.01**	**3.82**	**2.54**	**0.64**	**1.04**	2774	**1.95**
140N20-08	3511	705	**0.68**	3971	**0.84**	715	**0.47**	**4.90**	**10.76**	**0.03**	**1.18**	**0.89**	**0.64**	**0.66**	3573	1.87
140N20-09	3381	1001	**0.76**	4087	**0.91**	938	**0.24**	**2.91**	**13.60**	**0.14**	**61.74**	**42.29**	**8.71**	**32.08**	3399	1.96
160N20-00	3514	753	**0.93**	5608	**1.33**	772	**0.15**	**3.24**	**8.51**	**0.07**	**1.53**	**1.08**	**0.69**	**0.92**	3514	**2.58**
160N20-01	3805	1098	**0.92**	5781	**1.45**	975	**0.65**	**5.38**	**9.18**	**0.03**	**8.07**	**5.13**	**2.13**	**4.82**	3805	**2.48**
160N20-02	4136	1113	**0.90**	6310	**1.52**	966	**0.66**	**231.03**	**941.79**	**90.16**	**8.21**	**4.47**	**2.25**	**3.11**	4185	2.43
160N20-03	3847	956	**0.95**	5448	**1.27**	1007	**0.20**	**58.44**	**127.31**	**0.31**	**77.58**	**22.16**	**3.08**	**18.40**	3847	**2.61**
160N20-04	3359	790	**0.83**	5137	**1.28**	757	**0.64**	**0.51**	**6.78**	**0.06**	**4.53**	**2.76**	**1.50**	**2.39**	3359	**2.44**
160N20-06	3889	1070	**0.91**	6230	**1.70**	1031	**0.67**	**45.84**	**147.12**	**1.37**	**12.37**	**5.54**	**2.93**	**3.70**	3919	2.63
160N20-07	3922	934	**0.85**	5994	**1.37**	999	**0.64**	**47.72**	**97.48**	**0.31**	**3.47**	**1.38**	**0.75**	**1.05**	3960	2.31
160N20-08	3750	979	**1.02**	5972	**1.64**	1003	**0.82**	**6.22**	**37.05**	**0.24**	**78.10**	**66.41**	**53.07**	**60.34**	3752	2.66
160N20-09	3851	1205	**1.10**	5967	**1.70**	1129	**0.77**	**132.78**	**434.80**	**73.03**	**80.29**	**67.07**	**17.30**	**59.77**	3875	2.60
160N20-10	3684	713	**0.80**	5317	**1.16**	682	**0.18**	**0.12**	**0.13**	**0.01**	**1.10**	**0.85**	**0.44**	**0.80**	3684	**2.56**
180N20-00	4630	1168	**1.13**	7129	**2.12**	1118	**0.77**	**55.96**	**797.10**	**1.10**	**7.31**	**4.28**	**2.56**	**10.61**	4630	**3.30**
180N20-01	4551	976	**1.05**	7797	**1.91**	1000	**0.82**	**14.98**	**115.17**	**0.09**	**4.71**	**2.73**	**1.51**	**3.25**	4576	3.09
180N20-02	4554	1037	**1.02**	7494	**1.80**	999	**0.79**	**5.28**	**26.94**	**0.03**	**2.64**	**1.96**	**1.23**	**1.91**	4627	3.27
180N20-04	4200	1013	**1.05**	8115	**2.27**	984	**0.81**	**12.61**	**1178.21**	**0.06**	**5.25**	**3.35**	**2.52**	**2.84**	4200	**3.52**
180N20-07	4079	1005	**1.14**	6609	**1.93**	911	**0.82**	**5.44**	**11.41**	**0.16**	**6.81**	**3.07**	**1.32**	**4.27**	4079	**3.27**
180N20-08	4608	1303	**1.23**	8524	**1.95**	1390	**0.87**	**9.46**	**111.33**	**20.97**	**17.25**	**11.85**	**6.47**	**12.79**	4670	3.56
180N20-09	4664	1138	**1.42**	6695	**2.40**	1058	**1.02**	**5.80**	**8.78**	**0.04**	**11.21**	**6.15**	**2.33**	**2.51**	4664	**3.54**
180N20-10	4882	1380	**1.21**	7446	**1.83**	1250	**0.83**	**31.42**	**103.16**	**0.75**	**11.99**	**7.77**	**3.97**	**11.86**	4888	3.43
180N20-11	4487	1164	**1.15**	7533	**1.95**	1163	**0.79**	**7.53**	**16.40**	**1.08**	**16.49**	**9.33**	**4.13**	**5.17**	4522	2.97
180N20-12	4490	1029	**1.09**	7187	**1.83**	981	**0.81**	**1.82**	**12.25**	**0.07**	**2.82**	**2.00**	**0.98**	**1.25**	4507	3.26
Average		761.3	0.70	4493.6	1.03	735.6	0.43	18.24	105.71	3.88	12.11	7.81	3.22	6.92	–	1.98

Table 12 – Results for grid digraphs with $|V| = 9 \times 9$ and $N = 15$ (FERONE *et al.*, 2016).

Data set		(DN)		(FN)		$(FN)^+$		BBbf	BBdf	NewBB	NDLH			DLH	GRASP	
inst	*opt*	*iter*	*CPU*	*iter*	*CPU*	*iter*	*CPU*	*CPU*	*CPU*	*CPU*	CPU^+	$\overline{CPU}$	CPU^-	*CPU*	*UB*	*CPU*
N19t35-000	2393	419	**0.13**	1179	**0.08**	389	**0.05**	**0.03**	**0.18**	**0.01**	**0.34**	**0.16**	**0.07**	**0.08**	2393	**0.43**
N19t35-010	4200	741	**0.18**	1244	**0.12**	687	**0.10**	**2.52**	**89.47**	**0.06**	**1.11**	**0.87**	**0.29**	**0.69**	4200	**0.42**
N19t35-020	3271	606	**0.12**	1206	**0.09**	562	**0.08**	**0.49**	**0.82**	**0.05**	**0.43**	**0.34**	**0.17**	**0.32**	3289	0.42
N19t35-030	2535	495	**0.12**	1111	**0.08**	455	**0.08**	**0.07**	**0.09**	**0.01**	**0.34**	**0.27**	**0.18**	**0.35**	2535	**0.44**
N19t35-040	2980	558	**0.10**	1171	**0.09**	525	**0.07**	**0.46**	**43.63**	**0.32**	**0.48**	**0.32**	**0.20**	**0.31**	2980	**0.38**
N19t35-050	3011	609	**0.15**	1468	**0.12**	568	**0.12**	**116.48**	**201.93**	OM	**11.19**	**9.74**	**7.42**	**10.68**	3039	0.41
N19t35-060	2829	533	**0.17**	1194	**0.08**	480	**0.07**	**0.19**	**0.37**	**0.33**	**1.20**	**1.05**	**0.81**	**0.83**	2928	**0.46**
N19t35-070	2421	286	**0.09**	1085	**0.08**	266	**0.07**	**0.01**	**0.02**	**0.01**	**0.14**	**0.09**	**0.06**	**0.06**	2421	**0.38**
N19t35-080	2810	672	**0.13**	1100	**0.10**	665	**0.09**	**0.84**	**1.53**	**0.01**	**9.74**	**3.76**	**2.08**	**2.62**	2810	**0.46**
N19t35-090	2790	490	**0.12**	1197	**0.09**	465	**0.06**	**0.17**	**1.12**	**0.01**	**0.30**	**0.25**	**0.19**	**0.20**	2790	**0.38**
Average		533.3	0.13	1165.2	0.09	499.3	0.07	0.53	15.25	0.09	1.56	0.79	0.45	0.61	–	0.42

Table 13 – Results for grid digraphs with $|V| = 10 \times 10$ and $N = 19$ (FERONE *et al.*, 2016).

Data set		(DN)		(FN)		$(FN)^+$		BBbf	BBdf	NewBB	NDLH			DLH	GRASP	
inst	*opt*	*iter*	*CPU*	*iter*	*CPU*	*iter*	*CPU*	*CPU*	*CPU*	*CPU*	CPU^+	$\overline{CPU}$	CPU^-	*CPU*	*UB*	*CPU*
N19t35-00	4619	1396	**0.52**	2598	**0.35**	1311	**0.36**	OM	OM	**245.86**	**38.75**	**24.94**	**1.12**	**0.95**	4652	0.76
N19t35-10	4443	917	**0.17**	1882	**0.19**	836	**0.17**	**5.47**	**88.34**	**0.11**	**4.95**	**2.67**	**1.10**	**1.90**	4468	0.63
N19t35-20	4945	1229	**0.31**	2234	**0.27**	1135	**0.23**	OM	OM	OM	**6.74**	**4.14**	**2.56**	**2.18**	5253	0.72
N19t35-30	3591	608	**0.15**	1957	**0.22**	610	**0.14**	**0.35**	**6.90**	**0.01**	**0.42**	**0.29**	**0.15**	**0.22**	3591	**0.72**
N19t35-40	4064	1057	**0.37**	2146	**0.27**	946	**0.22**	**81.91**	**218.77**	**6.53**	**29.57**	**26.56**	**15.10**	**0.93**	4064	**0.60**
N19t35-50	3498	575	**0.12**	1944	**0.20**	557	**0.13**	**0.44**	**0.99**	**0.01**	**0.47**	**0.32**	**0.14**	**0.22**	3498	**0.60**
N19t35-60	4283	965	**0.14**	2092	**0.22**	946	**0.20**	OM	**1790.24**	**7.99**	**21.42**	**7.04**	**3.22**	**5.06**	4283	**0.71**
N19t35-70	4306	951	**0.17**	2244	**0.25**	916	**0.19**	OM	OM	**78.56**	**3.72**	**1.90**	**0.99**	**1.18**	4346	0.72
N19t35-80	4869	1144	**0.24**	2212	**0.24**	1122	**0.19**	OM	OM	OM	**21.71**	**18.61**	**6.28**	**18.04**	4908	0.75
N19t35-90	4585	1031	**0.26**	1972	**0.24**	978	**0.18**	**67.52**	**104.65**	**0.24**	**20.43**	**11.25**	**1.16**	**1.93**	4604	0.70
Average		837.6	0.22	1980.2	0.22	785.4	0.17	31.14	83.93	1.38	11.17	8.22	3.53	1.04	–	0.65

Table 14 – Results for grid digraphs with $|V| = 5 \times 20$ and $N = 19$ (FERONE *et al.*, 2016).

Data set		(DN)		(FN)		$(FN)^+$		BBbf	BBdf	NewBB	NDLH			DLH	GRASP	
inst	*opt*	*iter*	*CPU*	*iter*	*CPU*	*iter*	*CPU*	*CPU*	*CPU*	*CPU*	CPU^+	$\overline{CPU}$	CPU^-	*CPU*	*UB*	*CPU*
N19t35-00	4606	912	**0.17**	2042	**0.18**	943	**0.15**	**44.44**	**475.92**	**42.66**	**20.06**	**9.35**	**2.13**	**3.55**	4643	0.84
N19t35-10	6094	1198	**0.28**	1764	**0.20**	1414	**0.20**	OM	OM	OM	**1.41**	**0.74**	**0.45**	**0.62**	6214	0.72
N19t35-20	4827	1168	**0.20**	2252	**0.20**	1066	**0.20**	OM	OM	OM	**21.94**	**15.06**	**2.92**	**18.07**	4948	0.72
N19t35-30	3939	739	**0.13**	2026	**0.15**	653	**0.14**	**17.03**	**23.65**	**0.71**	**1.85**	**1.13**	**0.50**	**0.66**	3947	0.56
N19t35-40	5632	1639	**0.47**	2526	**0.41**	1520	**0.30**	OM	OM	OM	**48.12**	**14.25**	**1.08**	**1.07**	5803	0.67
N19t35-50	4433	1193	**0.30**	2470	**0.22**	1044	**0.19**	OM	OM	OM	**3.76**	**2.26**	**0.68**	**2.95**	4433	**0.62**
N19t35-60	5247	1257	**0.19**	2165	**0.22**	1241	**0.20**	OM	OM	OM	**12.84**	**5.86**	**3.14**	**3.85**	5395	0.7
N19t35-70	4949	1037	**0.16**	2044	**0.16**	1024	**0.19**	OM	OM	**15.82**	**20.54**	**5.37**	**0.83**	**1.15**	4965	0.71
N19t35-80	5656	1623	**0.49**	2966	**0.37**	1405	**0.30**	OM	OM	OM	**37.42**	**18.05**	**0.96**	**1.06**	5841	0.54
N19t35-90	4736	949	**0.19**	1792	**0.18**	886	**0.18**	**0.34**	OM	**0.05**	**1.10**	**0.84**	**0.48**	**0.57**	4789	0.66
Average		825.5	0.15	2034.0	0.16	798.0	0.15	30.74	249.79	21.69	10.95	5.24	1.32	2.10	–	0.70

Table 15 – Results for grid digraphs with $|V| = 7 \times 15$ and $N = 19$ (FERONE *et al.*, 2016).

Data set		(DN)		(FN)		$(FN)^+$		BBbf	BBdf	NewBB	NDLH			DLH	GRASP	
inst	*opt*	*iter*	*CPU*	*iter*	*CPU*	*iter*	*CPU*	*CPU*	*CPU*	*CPU*	CPU^+	$\overline{CPU}$	CPU^-	*CPU*	*UB*	*CPU*
N19t35-00	4546	1108	**0.27**	2111	**0.23**	1015	**0.21**	OM	OM	**0.45**	**1.61**	**0.68**	**0.33**	**0.53**	4546	**1.00**
N19t35-10	5198	1223	**0.21**	1911	**0.19**	1278	**0.19**	OM	OM	**45.01**	**2.67**	**1.65**	**0.769**	**1.17**	5233	0.79
N19t35-20	4174	918	**0.29**	1992	**0.31**	857	**0.27**	**113.17**	**142.89**	**1.80**	**29.50**	**23.95**	**14.64**	**17.78**	4232	0.62
N19t35-30	3479	643	**0.16**	1830	**0.17**	608	**0.12**	**0.18**	**0.21**	**0.01**	**0.77**	**0.44**	**0.28**	**0.28**	3479	**0.64**
N19t35-40	4242	689	**0.22**	1828	**0.21**	655	**0.19**	**0.20**	**34.43**	**0.03**	**0.63**	**0.39**	**0.17**	**0.55**	4242	**0.75**
N19t35-50	3688	730	**0.14**	2075	**0.17**	687	**0.15**	**35.39**	**119.40**	**0.34**	**0.88**	**0.58**	**0.33**	**0.38**	3700	0.83
N19t35-60	4501	836	**0.15**	1975	**0.16**	830	**0.14**	**6.49**	**10.13**	**0.06**	**6.79**	**2.14**	**0.54**	**0.85**	4501	**0.95**
N19t35-70	3500	628	**0.14**	1992	**0.15**	647	**0.11**	**1.43**	**3.39**	**0.03**	**0.88**	**0.53**	**0.25**	**0.76**	3500	**0.74**
N19t35-80	5513	1209	**0.39**	2338	**0.28**	1182	**0.24**	OM	OM	OM	**25.00**	**20.09**	**16.84**	**20.59**	5551	0.84
N19t35-90	3586	673	**0.16**	1757	**0.17**	624	**0.17**	**1.28**	**4.83**	**0.01**	**34.43**	**19.09**	**2.57**	**17.08**	3586	**0.67**
Average		731.0	0.18	1921.2	0.19	701.1	0.16	22.59	45.04	0.33	10.55	6.73	2.68	5.38	–	0.74

Table 16 – Results for grid digraphs with $|V| = 25 \times 4$ and $N = 19$ (FERONE *et al.*, 2016).

Data set		(DN)		(FN)		$(FN)^+$		BBbf	BBdf	NewBB	NDLH			DLH	GRASP	
inst	*opt*	*iter*	*CPU*	*iter*	*CPU*	*iter*	*CPU*	*CPU*	*CPU*	*CPU*	CPU^+	$\overline{CPU}$	CPU^-	*CPU*	*UB*	*CPU*
N19t35-00	5754	1528	**0.29**	2265	**0.23**	1492	**0.19**	OM	OM	OM	**22.78**	**16.34**	**2.57**	**20.03**	5834	0.94
N19t35-10	5382	1049	**0.28**	1927	**0.21**	976	**0.17**	OM	OM	OM	**7.63**	**4.95**	**1.92**	**1.48**	5751	1.13
N19t35-20	6178	1655	**0.32**	2685	**0.26**	1507	**0.22**	OM	OM	OM	**22.86**	**20.29**	**16.61**	**20.02**	6384	0.94
N19t35-30	3779	693	**0.41**	1926	**0.22**	565	**0.23**	**1.86**	**3.91**	**0.02**	**17.78**	**14.01**	**2.52**	**9.01**	3779	**1.00**
N19t35-40	6921	1272	**0.28**	2020	**0.22**	1351	**0.20**	OM	OM	OM	**21.88**	**18.68**	**15.13**	**18.29**	7121	0.96
N19t35-50	5660	1412	**0.55**	2217	**0.25**	1354	**0.26**	OM	OM	OM	**22.46**	**19.39**	**15.70**	**19.13**	5754	1.04
N19t35-60	6406	1197	**0.27**	1948	**0.21**	1218	**0.18**	OM	OM	OM	**15.10**	**3.53**	**1.19**	**1.43**	6739	0.89
N19t35-70	4985	962	**0.27**	1788	**0.19**	1028	**0.16**	**1.25**	**43.88**	**0.56**	**1.22**	**1.00**	**0.69**	**0.64**	4985	**1.32**
N19t35-80	5391	1362	**0.31**	2287	**0.22**	1298	**0.19**	OM	OM	OM	**22.74**	**14.75**	**2.14**	**3.01**	5615	1.19
Average		827.5	0.34	1857.0	0.21	796.5	0.19	1.56	23.90	0.29	9.50	7.51	1.60	4.83	–	1.16

Table 17 – Results for grid digraphs with $|V| = 12 \times 12$ and $N = 27$ (FERONE *et al.*, 2016).

Data set		(DN)			(FN)			$(FN)^+$			BBbf	BBdf	NewBB	NDLH				DLH	GRASP	
inst	*opt*	*bb*	*iter*	*CPU*	*bb*	*iter*	*CPU*	*bb*	*iter*	*CPU*	*CPU*	*CPU*	*CPU*	CPU^+	$\overline{CPU}$	CPU^-		*CPU*	*UB*	*CPU*
N19t35-00	7017	0	2355	**0.86**	0	5428	**0.88**	0	2135	**0.51**	OM	OM	OM	**3600.00**	**1142.29**	**37.07**		**44.60**	7091	1.70
N19t35-10	8752	4	3332	**3.07**	3	6956	**3.62**	6	4126	**3.83**	OM	OM	OM	**3600.00**	**3600.00**	**39.53**		**50.38**	9027	1.59
N19t35-20	5445	0	1314	**0.60**	0	4462	**0.69**	0	1310	**0.41**	OM	OM	**13.12**	**4.31**	**2.58**	**1.49**		**2.13**	5445	**1.50**
N19t35-30	8530	0	3019	**0.94**	0	7760	**1.64**	0	2897	**0.70**	**147.35**	OM	OM	**177.19**	**137.08**	**36.24**		**44.01**	8780	1.98
N19t35-40	6346	0	1478	**0.51**	0	3954	**0.58**	0	1390	**0.40**	**11.71**	**65.12**	**0.44**	**3.63**	**2.55**	**1.15**		**1.20**	6346	1.48
N19t35-50	6261	0	1910	**0.61**	0	5906	**0.83**	0	1943	**0.41**	OM	OM	OM	**28.15**	**15.23**	**5.60**		**9.00**	6381	1.65
N19t35-60	6359	0	1775	**0.53**	0	5601	**0.78**	0	1741	**0.35**	OM	OM	OM	**14.21**	**8.25**	**2.96**		**3.13**	6432	1.62
N19t35-70	6185	0	1905	**0.47**	0	5843	**0.83**	0	1642	**0.35**	OM	OM	OM	**6.75**	**4.59**	**1.94**		**3.50**	6535	1.45
N19t35-80	7210	0	2362	**0.74**	0	5572	**0.86**	0	2248	**0.52**	OM	OM	OM	**54.96**	**36.23**	**7.47**		**16.44**	7380	1.71
N19t35-90	6611	2	2348	**3.07**	3	6290	**2.26**	0	2382	**1.09**	OM	OM	OM	**81.88**	**68.56**	**33.26**		**39.98**	6786	1.47
Average		0	1478.0	0.51	0	3954.0	0.58	0	1390.0	0.40	11.71	65.12	0.44	3.63	2.55	1.15		1.20	–	1.48

Table 18 – Results for new random digraphs with $|V| = 100$.

Data set			(DN)			(FN)			$(FN)^+$			NDLH				DLH		GRASP	
$\|A\|$	N	opt	bb	$iter$	CPU	bb	$iter$	CPU	bb	$iter$	CPU	UB	CPU^+	$\overline{CPU}$	CPU^-	UB	CPU	UB	CPU
6608	35	191	0	2441	**5.57**	0	6686	**17.02**	0	2578	**3.57**	191	**321.00**	**74.90**	**33.01**	191	**77.72**	206	3.97
6625	41	203	0	2184	**12.39**	3	5860	**27.38**	1	1946	**13.80**	203	**436.59**	**428.47**	**401.33**	203	**298.55**	212	10.04
6571	42	264	0	1553	**3.29**	0	6330	**13.60**	0	1664	**2.54**	264	**265.87**	**168.37**	**133.85**	264	**103.11**	275	6.89
6566	44	264	0	4783	**20.73**	2	13378	**51.84**	0	4730	**21.71**	264	**490.66**	**453.31**	**448.66**	265	444.76	283	6.50
6551	48	371	0	3228	**18.28**	0	10804	**28.69**	0	3590	**16.12**	371	**165.88**	**132.52**	**107.31**	372	508.06	388	5.13
6577	49	322	0	2895	**4.48**	0	12251	**29.27**	0	2225	**2.93**	322	**90.56**	**65.84**	**46.75**	322	**41.24**	359	7.01
6600	50	281	0	2880	**14.72**	0	10407	**29.50**	0	2815	**6.03**	281	**377.65**	**263.28**	**103.18**	282	420.80	295	8.92
6626	53	353	0	4481	**27.71**	0	16087	**43.97**	3	4393	**28.21**	353	**593.91**	**571.62**	**555.15**	354	595.46	387	4.88
6573	54	314	34	11094	**59.38**	20	31304	**145.02**	35	12019	**59.33**	314	**613.71**	**595.36**	**577.05**	319	591.40	347	6.29
6602	56	390	55	12898	**59.50**	59	30727	**141.52**	47	9448	**47.91**	403	638.81	612.60	592.62	407	606.13	438	6.29
6663	65	506	0	6940	**46.11**	4	20783	**119.30**	14	5571	**43.55**	506	**744.07**	**718.04**	**687.54**	512	701.08	554	7.60
6549	67	588	29	8286	**52.54**	10	23594	**113.37**	9	7343	**42.26**	588	**745.91**	**725.51**	**692.06**	593	705.53	653	5.24
6640	68	542	50	18349	**76.57**	13	23631	**135.14**	12	7413	**47.42**	542	**767.29**	**753.29**	**717.39**	544	727.74	685	3.77
6546	69	609	0	6890	**38.10**	0	22072	**97.98**	1	7256	**46.03**	609	**766.46**	**742.91**	**702.23**	616	715.97	680	4.45
6609	78	653	3	10055	**54.83**	3	29282	**151.17**	1	9597	**49.91**	653	**876.49**	**856.27**	**839.00**	665	841.03	790	3.95
6588	80	612	24	21363	**202.98**	3	33668	**233.83**	22	21765	**143.27**	612	**899.07**	**878.48**	**867.18**	618	874.08	779	5.12
6586	81	696	99	25740	**130.45**	126	74623	**387.13**	149	24620	**110.46**	696	**924.52**	**901.29**	**864.74**	717	911.89	846	4.42
6580	84	743	27	19999	**150.84**	109	127194	**900.93**	42	24047	**147.46**	743	**965.71**	**942.31**	**894.33**	752	941.02	924	4.22
6595	85	675	10	13064	**128.34**	32	35194	**321.07**	4	10188	**75.01**	675	**960.22**	**923.87**	**882.30**	677	934.55	796	4.80
6655	93	858	75	43042	**231.70**	115	53964	**423.56**	51	22880	**155.43**	858	**1080.81**	**1059.55**	**1012.36**	869	1058.22	1072	4.73
Average			20.3	11108.2	66.92	24.9	29391.9	170.56	19.5	9304.4	53.15	–	636.26	593.39	557.90	–	604.92	–	5.71

Table 19 – Results for new random digraphs with $|V| = 150$.

Data set			(DN)			(FN)			$(FN)^+$			NDLH				DLH		GRASP	
$\|A\|$	N	opt	bb	$iter$	CPU	bb	$iter$	CPU	bb	$iter$	CPU	UB	CPU^+	$\overline{CPU}$	CPU^-	UB	CPU	UB	CPU
14927	32	53	0	4649	**43.67**	0	11079	**66.28**	0	5051	**34.28**	53	**959.73**	**933.08**	**852.91**	56	829.30	58	26.97
14903	37	88	10	5842	**71.80**	26	16784	**115.61**	24	6283	**69.49**	88	**1138.59**	**1107.00**	**1062.60**	90	1114.69	96	23.83
14969	39	108	12	4784	**55.37**	42	16608	**114.17**	18	3872	**44.74**	108	**1212.64**	**1179.62**	**1130.74**	115	1158.32	115	31.49
14871	40	99	0	2790	**12.11**	0	8127	**54.70**	0	2648	**10.28**	99	**844.36**	**449.15**	**180.85**	99	**915.77**	105	21.39
15041	44	121	0	6276	**58.88**	2	17210	**218.33**	0	6130	**41.83**	123	**1392.45**	**1352.72**	**1273.21**	127	1340.28	143	27.47
14889	45	119	86	44013	**351.25**	5	20095	**209.57**	5	6603	**84.08**	126	1416.89	1381.21	1342.14	133	1382.85	134	29.93
14925	51	173	61	34424	**251.97**	52	44348	**499.70**	87	39407	**454.00**	173	**1633.65**	**1569.39**	**1515.26**	177	1573.70	207	13.87
14955	52	229	0	7543	**81.70**	0	21457	**253.71**	0	7065	**89.92**	229	**1662.43**	**1351.52**	**473.46**	231	1606.36	250	15.84
14828	54	161	6	10140	**138.98**	11	25344	**289.56**	5	9563	**117.85**	161	**1707.38**	**1625.42**	**1585.49**	162	1602.34	186	27.84
14877	55	242	0	7345	**57.23**	0	20331	**241.17**	0	7454	**39.45**	242	**1715.10**	**1054.14**	**365.32**	243	1661.22	274	14.23
14854	56	177	93	45806	**797.64**	35	51430	**1219.35**	128	61295	**640.80**	185	1765.72	1732.46	1699.87	189	1730.76	211	28.46
14879	59	236	0	9341	**128.98**	0	28625	**281.13**	0	9136	**109.51**	238	**1848.18**	**1812.19**	**1772.49**	243	1792.22	270	17.69
14937	60	211	138	74256	**1031.41**	169	169133	**2707.07**	91	42211	**652.28**	219	1885.30	1849.94	1798.85	224	1831.87	243	28.89
14890	63	282	32	22345	**493.79**	29	62208	**739.88**	16	13238	**146.10**	282	**1951.06**	**1883.54**	**1793.95**	284	1826.58	325	19.07
14943	64	230	93	57712	**354.08**	119	156014	**2482.93**	86	42601	**347.24**	234	2076.63	1935.42	1872.78	238	1876.39	267	24.59
14892	75	339	56	46292	**377.37**	34	63880	**954.48**	7	16111	**248.30**	339	**2619.85**	**2508.23**	**2445.65**	349	2520.16	407	18.59
14908	79	453	89	33196	**472.52**	50	70976	**1130.65**	59	28060	**415.61**	453	**2736.31**	**2701.92**	**2604.51**	465	2674.08	550	21.03
14903	87	382	252	202306	**3600.45**	298	446229	**3203.00**	907	336293	**3181.69**	396	3001.92	2938.66	2860.45	408	2926.96	486	25.32
14943	99	441	40	67165	**664.58**	63	85545	**1818.78**	15	42734	**602.56**	449	3528.15	3461.53	3397.23	462	3448.77	572	28.43
14975	106	536	298	166232	**1715.71**	231	295507	**3136.42**	435	289442	**1716.66**	556	3600.00	3598.84	3588.41	565	3600.00	–	OM
Average			50.9	36117.1	475.99	49.2	70285.4	873.69	76.2	36092.3	385.79	–	1847.18	1727.74	1580.41	–	1779.61	–	23.42

4 FINAL REMARKS AND FUTURE WORK

In this work, we investigate two NP-Hard routing problems: the shortest path with negative cycles (SPNC) and the constrained shortest path tour problem (CSPTP). For the SPNC, we propose three exact solution approches, namely, a compact mixed integer linear programming model, a specialized branch-and-bound (B&B) algorithm, and a cutting-plane (CP) method. For the CSPTP, we introduce both pure integer and mixed integer linear programming models, as well as deterministic and non-deterministic Lagrangian heuristics. Our SPNC and CSPTP contributions are all tested on randomly generated and benchmark instances from the literature.

Regarding the SPNC, we show that our three exact solution approaches outperform existing state-of-the-art mathematical programming models when achieving optimal solutions in smaller execution times. For benchmark instances consisting of general digraphs, our compact model alternates better results with our B&B algorithm. For randomly generated instances, B&B alternates better results with CP. For benchmark grid digraphs, our model performs better than other approaches.

On what concerns the CSPTP, we show that our exact models, in general, outperform B&B methods from the literature and we also show the impact of the valid inequalities. The execution time is strongly reduced when taking them into account, mainly for randomly generated digraphs. It is worth emphasizing the quality of the Lagrangian heuristics in terms of effectiveness and efficiency when achieving the optimal solution for some instances in smaller execution time when compared to exact approaches. For randomly generated instances, we show that state-of-the-art B&B do not find any optimal solution within one hour of execution time, while our exact contributions do.

An interesting line for future research is to study real-life problems where negative costs on the arcs of the digraph arise. The proposed exact approaches for the SPNC are easy-to-implement and could be adapted to tackle combinatorial optimization problems with such a particular characteristic. Another future work is to investigate decomposition methods for the CSPTP. Preliminary experiments proved that omitting nodes from clusters tends to reduce the execution time. The choice for which nodes to omit is intrinsically related to decomposition.

Lastly but not least, the application of metaheuristics (e.g., Genetic Algorithm) seems to be a natural way to find promising feasible solutions for huge SPNC and CSPTP instances, which also deserves our attention.

REFERENCES

ANDRADE, R. C.; ARAUJO, K. A. G.; SARAIVA, R. D. Um método primal-dual para o problema do caminho mínimo em digrafos na presença de ciclos absorventes. **XLVII Simpósio Brasileiro de Pesquisa Operacional**, [*s.n.*], [*s.l.*], p. 2904–2911 (in Portuguese), 2015.

ANDRADE, R. C.; FREITAS, A. T. Disjunctive combinatorial branch in a subgradient tree algorithm for the dcmst problem with vns-lagrangian bounds. **Electronic Notes in Discrete Mathematics**, Elsevier, [*s.l.*], v. 41, p. 5–12, 2013.

ANDRADE, R. C.; SARAIVA, R. D. Elementary shortest paths avoiding negative circuits. **18th Latin-Iberoamerican Conference on Operations Research**, [*s.n.*], [*s.l.*], p. 30–37, 2016.

ANDRADE, R. C.; SARAIVA, R. D. MTZ-primal-dual model, cutting-plane, and combinatorial branch-and-bound for shortest paths avoiding negative cycles. **Annals of Operations Research**, Springer, [*s.l.*], p. 1–26, 2017.

ANDRADE, R. C.; SARAIVA, R. D. An integer linear programming model for the constrained shortest path tour problem. **Electronic Notes in Discrete Mathematics**, [*s.n.*], [*s.l.*], v. 69, p. 141 – 148, 2018. ISSN 1571-0653. Joint EURO/ALIO International Conference 2018 on Applied Combinatorial Optimization (EURO/ALIO 2018).

BATSYN, M.; GOLDENGORIN, B.; KOCHETUROV, A.; PARDALOS, P. M. Tolerance-based vs. cost-based branching for the asymmetric capacitated vehicle routing problem. In: **Models, Algorithms, and Technologies for Network Analysis**. [*s.l.*]: Springer, 2013. p. 1–10.

BEASLEY, J. E.; CHRISTOFIDES, N. An algorithm for the resource constrained shortest path problem. **Networks**, Wiley Online Library, [*s.l.*], v. 19, n. 4, p. 379–394, 1989.

BERTOSSI, A. A. The edge Hamiltonian path problem is NP-Complete. **Information Processing Letters**, Elsevier, [*s.l.*], v. 13, n. 4-5, p. 157–159, 1981.

BONDY, J. A.; MURTY, U. S. R. *et al.* **Graph theory with applications**. [*s.l.*]: Citeseer, 2008. v. 290.

DESROCHERS, M.; LAPORTE, G. Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. **Operations Research Letters**, Elsevier, [*s.l.*], v. 10, n. 1, p. 27–36, 1991.

DIJKSTRA, E. W. A note on two problems in connexion with graphs. **Numerische mathematik**, Springer, [*s.l.*], v. 1, n. 1, p. 269–271, 1959.

DREXL, M. A note on the separation of subtour elimination constraints in elementary shortest path problems. **European Journal of Operational Research**, Elsevier, [*s.l.*], v. 229, n. 3, p. 595–598, 2013.

FERONE, D.; FESTA, P.; GUERRIERO, F. An efficient exact approach for the constrained shortest path tour problem. **Optimization Methods and Software**, Taylor & Francis, [*s.l.*], p. 1–20, 2019.

FERONE, D.; FESTA, P.; GUERRIERO, F.; LAGANÀ, D. The constrained shortest path tour problem. **Computers & Operations Research**, Elsevier, [*s.l.*], v. 74, p. 64–77, 2016.

FESTA, P. Complexity analysis and optimization of the shortest path tour problem. **Optimization Letters**, Springer, [*s.l.*], v. 6, n. 1, p. 163–175, 2012.

FESTA, P.; GUERRIERO, F.; LAGANÀ, D.; MUSMANNO, R. Solving the shortest path tour problem. **European Journal of Operational Research**, Elsevier, [*s.l.*], v. 230, n. 3, p. 464–474, 2013.

FESTA, P.; PALLOTTINO, S. **A pseudo-random networks generator**. [*s.l.*], 2003.

FLOYD, R. W. Algorithm 97: shortest path. **Communications of the ACM**, ACM, [*s.l.*], v. 5, n. 6, p. 345, 1962.

FORD, L. R. **Network flow theory**. [*s.l.*], 1956.

FREDMAN, M. L.; TARJAN, R. E. Fibonacci heaps and their uses in improved network optimization algorithms. **Journal of the ACM (JACM)**, ACM, [*s.l.*], v. 34, n. 3, p. 596–615, 1987.

GAREY, M. R.; JOHNSON, D. S. **Computers and intractability**. [*s.l.*]: wh freeman New York, 2002. v. 29.

GEOFFRION, A. M. Lagrangian relaxation and its uses in integer programming. **Mathematical Programming Study**, [*s.n.*], [*s.l.*], v. 2, 1974.

GHOUILA-HOURI, A. Caractérisation des matrices totalement unimodulaires. **Comptes Redus Hebdomadaires des Séances de l'Académie des Sciences (Paris)**, [*s.n.*], [*s.l.*], v. 254, p. 1192–1194, 1962.

GU, X.; MADDURI, K.; SUBRAMANI, K.; LAI, H. Improved algorithms for detecting negative cost cycles in undirected graphs. In: **Frontiers in Algorithmics**. [*s.l.*]: Springer, 2009. p. 40–50.

HANDLER, G. Y.; ZANG, I. A dual algorithm for the constrained shortest path problem. **Networks**, Wiley Online Library, [*s.l.*], v. 10, n. 4, p. 293–309, 1980.

HAOUARI, M.; MACULAN, N.; MRAD, M. Enhanced compact models for the connected subgraph problem and for the shortest path problem in digraphs with negative cycles. **Computers & Operations Research**, Elsevier, [*s.l.*], v. 40, n. 10, p. 2485–2492, 2013.

HEEGAARD, P. E.; TRIVEDI, K. S. Network survivability modeling. **Computer Networks**, Elsevier, [*s.l.*], v. 53, n. 8, p. 1215–1234, 2009.

HOUGARDY, S. The floyd–warshall algorithm on graphs with negative cycles. **Information Processing Letters**, Elsevier, [*s.l.*], v. 110, n. 8, p. 279–281, 2010.

IBRAHIM, M. S. **Etude de formulations et inégalités valides pour le problème du plus court chemin dans les graphes avec des circuits absorbants**. Tese (Doutorado) — Université Pierre et Marie Curie, Paris, France, [*s.l.*], 2007.

IBRAHIM, M. S. A strong class of lifted valid inequalities for the shortest path problem in digraphs with negative cost cycles. **Journal of Mathematics Research**, [*s.n.*], [*s.l.*], v. 7, n. 4, p. 162–166, 2015.

IBRAHIM, M. S.; MACULAN, N.; MINOUX, M. A strong flow-based formulation for the shortest path problem in digraphs with negative cycles. **International Transactions in Operational Research**, Wiley Online Library, [*s.l.*], v. 16, n. 3, p. 361–369, 2009.

IBRAHIM, M. S.; MACULAN, N.; MINOUX, M. **Le Problème Du Plus Court Chemin Avec Des Longueurs Negatives**. [*s.l.*]: Editions universitaires européennes, Saarbrucken (Allemagne), 2015.

IBRAHIM, M. S.; MACULAN, N.; MINOUX, M. Valid inequalities and lifting procedures for the shortest path problem in digraphs with negative cycles. **Optimization Letters**, Springer, [*s.l.*], v. 9, n. 2, p. 345–357, 2015.

IBRAHIM, M. S.; MACULAN, N.; OUZIA, H. An efficient cutting plane algorithm for the minimum weighted elementary directed cycle problem in planar digraphs. **RAIRO-Operations Research**, EDP Sciences, [*s.l.*], v. 50, n. 3, p. 665–675, 2016.

JOHNSON, D. B. Efficient algorithms for shortest paths in sparse networks. **Journal of the ACM (JACM)**, ACM, [*s.l.*], v. 24, n. 1, p. 1–13, 1977.

JOKSCH, H. C. The shortest route problem with constraints. **Journal of Mathematical analysis and applications**, Elsevier, [*s.l.*], v. 14, n. 2, p. 191–197, 1966.

LIM, C.; SMITH, J. C. Algorithms for discrete and continuous multicommodity flow network interdiction problems. **IIE Transactions**, Taylor & Francis, [*s.l.*], v. 39, n. 1, p. 15–26, 2007.

MEHLHORN, K.; PRIEBE, V.; SCHÄFER, G.; SIVADASAN, N. All-pairs shortest-paths computation in the presence of negative cycles. **Information Processing Letters**, Elsevier, [*s.l.*], v. 81, n. 6, p. 341–343, 2002.

MILLER, C. E.; TUCKER, A. W.; ZEMLIN, R. A. Integer programming formulation of traveling salesman problems. **Journal of the ACM (JACM)**, ACM, [*s.l.*], v. 7, n. 4, p. 326–329, 1960.

PUGLIESE, L. P.; GUERRIERO, F. On the shortest path problem with negative cost cycles. **Computational Optimization and Applications**, Springer, [*s.l.*], v. 63, n. 2, p. 559–583, 2016.

SARAIVA, R. D.; ANDRADE, R. C. Costrained shortest path tour problem: integer programming models, valid inequalities and lagrangian heuristics. **International Transactions in Operational Research**, Wiley Online Library, [*s.l.*], submitted, 2019.

SCHRIJVER, A. **Theory of linear and integer programming**. [*s.l.*]: John Wiley & Sons, 1998.

SHERALI, H. D.; ADAMS, W. P. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. **SIAM Journal on Discrete Mathematics**, SIAM, [*s.l.*], v. 3, n. 3, p. 411–430, 1990.

SRIMANI, P. K.; SINHA, B. P. Impossible pair constrained test path generation in a program. **Information Sciences**, Elsevier, [*s.l.*], v. 28, n. 2, p. 87–103, 1982.

SUBRAMANI, K. A zero-space algorithm for negative cost cycle detection in networks. **Journal of Discrete Algorithms**, Elsevier, [*s.l.*], v. 5, n. 3, p. 408–421, 2007.

SUBRAMANI, K.; KOVALCHICK, L. A greedy strategy for detecting negative cost cycles in networks. **Future Generation Computer Systems**, Elsevier, [*s.l.*], v. 21, n. 4, p. 607–623, 2005.

TACCARI, L. Integer programming formulations for the elementary shortest path problem. **European Journal of Operational Research**, Elsevier, [*s.l.*], v. 252, n. 1, p. 122–130, 2016.

YAMADA, T.; KINOSHITA, H. Finding all the negative cycles in a directed graph. **Discrete Applied Mathematics**, Elsevier, [*s.l.*], v. 118, n. 3, p. 279–291, 2002.

ANNEX A – SUPPLEMENTARY MATERIAL CONCERNING THE SHORTEST PATH WITH NEGATIVE CYCLES

Table 20 shows an example of a coefficient matrix, with respect to (CPD-CR), for a complete digraph with $|V| = 4$. The first four lines present coefficients from flow conservation constraints (2.2), (2.3), (2.14), while the last two lines present coefficients from degree constraints (2.15). If I contains only rows from flow conservation, then $I_1 \leftarrow I$, $I_2 = \emptyset$, and we are done. Now, let us suppose that I contains rows from degree constraints. Note that row 5 (resp. 6) contains only positive values from row 2 (resp. 3). As row 2 (resp. 3), when belonging to I, is present in I_1, then we have to add row 5 (resp. 6) to I_2. Otherwise, we could have a sum of value two if both rows 2 and 5 (resp. 3 and 6) belonged to I_1. The same reasoning is generalized for any digraph. The row from the degree constraint related to any vertex $i \in V - \{s,t\}$ is added to I_2 if and only if the row from the flow conservation constraint related to i is in I_1. Otherwise, the former row is added to I_1 as no other row has positive values in the same columns.

Table 20 – Example of (CPD-CR) coefficient matrix for a complete digraph with $|V| = 4$.

Node/Arc	x_{12}	x_{13}	x_{14}	x_{21}	x_{23}	x_{24}	x_{31}	x_{32}	x_{34}	x_{41}	x_{42}	x_{43}
1	−1	−1	−1	1	0	0	1	0	0	1	0	0
2	1	0	0	−1	−1	−1	0	1	0	0	1	0
3	0	1	0	0	1	0	−1	−1	−1	0	0	1
4	0	0	1	0	0	1	0	0	1	−1	−1	−1
2	1	0	0	0	0	0	0	1	0	0	1	0
3	0	1	0	0	1	0	0	0	0	0	0	1

ANNEX B – SUPPLEMENTARY MATERIAL CONCERNING THE CONSTRAINED SHORTEST PATH TOUR PROBLEM

A generic system matrix $A_{L(\mu)}$ is as follows.

$$
\begin{array}{c}
\overbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}^{\text{sub-path arc variables}} \quad \overbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}^{\text{frontier-node variables}} \\
\begin{array}{cccccccc|cccccccc}
x^1 & x^2 & \cdots & x^{k-1} & x^k & \cdots & x^{N-2} & x^{N-1} & y(T_1) & y(T_2) & \cdots & y(T_k) & \cdots & y(T_{N-1}) & y(T_N) \\
\hline
\mathscr{N} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \begin{pmatrix} -I_1 \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{pmatrix} & \begin{pmatrix} \mathbf{0} \\ +I_2 \\ \vdots \\ \mathbf{0} \end{pmatrix} & \cdots & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathscr{N} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \begin{pmatrix} \mathbf{0} \\ -I_2 \\ \vdots \\ \mathbf{0} \end{pmatrix} & \cdots & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\
\vdots & \vdots & \ddots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \vdots & \\
\mathbf{0} & \mathbf{0} & \cdots & \mathscr{N} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \begin{pmatrix} \mathbf{0} \\ \vdots \\ +I_k \\ \mathbf{0} \end{pmatrix} & \cdots & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathscr{N} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \begin{pmatrix} \mathbf{0} \\ \vdots \\ -I_k \\ \mathbf{0} \end{pmatrix} & \cdots & \mathbf{0} & \mathbf{0} \\
\vdots & \vdots & \cdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \ddots & \vdots & \vdots \\
\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathscr{N} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \begin{pmatrix} \mathbf{0} \\ \vdots \\ +I_{N-1} \\ \mathbf{0} \end{pmatrix} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathscr{N} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \begin{pmatrix} \mathbf{0} \\ \vdots \\ -I_{N-1} \\ \mathbf{0} \end{pmatrix} & \begin{pmatrix} \mathbf{0} \\ \vdots \\ \mathbf{0} \\ +I_N \end{pmatrix} \\
\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & [1\cdots1] & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & [1\cdots1] & \cdots & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\
\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & [1\cdots1] & \cdots & \mathbf{0} & \mathbf{0} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & [1\cdots1] & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \mathbf{0} & [1\cdots1]
\end{array}
\end{array}
$$

where $\mathscr{N}$ represents the network node-arc incidence matrix, I_k is an induced identity matrix of order $|T_k|$ associated with the frontier nodes of T_k, and $[1\cdots1]$ is a row vector of $|T_k|$ coefficients all equal to 1 related to the cardinality constraints (3.11), $k = 1,\ldots,N$. Note that the first

$1,\cdots,|V|,|V|+1,\cdots,2|V|,\ldots,(N-2)|V|+1,\cdots,(N-1)|V|$ rows of matrix $A_{L(\mu)}$ provide $N-1$ blocks of independent systems of flow conservation constraints determined based on the network node-arc incidence matrix $\mathscr{N}$, each one related to an elementary path between two consecutive sets of frontier nodes of the given order of visit of the T_k's. The last N rows of $A_{L(\mu)}$ are cardinality constraints, one for each T_k, $k=1,\ldots,N$, whose non-null coefficients of these rows are all equal to 1, associated with variables y of nodes in T_k. Every column of $A_{L(\mu)}$ associated with the x-variables has exactly two non-null coefficients, $+1$ and -1. Every column associated with the y-variables has at most three non-null coefficients, $+1$ and -1 in rows related to flow conservation constraints, and $+1$ in the row associated with the cardinality constraint of its corresponding frontier node set.

As an example, consider a complete digraph with four vertices. Let $N=3$, $s=1$ and $t=4$. Furthermore, let $T_1=\{1\}$, $T_2=\{2,3\}$ and $T_3=\{4\}$. The coefficient matrix $A_{L(\mu)}$ associated with the example is in Table 21.

Table 21 – Example of a coefficient matrix $A_{L(\mu)}$.

Row/Variable	x^1_{12}	x^1_{13}	x^1_{14}	x^1_{21}	x^1_{23}	x^1_{24}	x^1_{31}	x^1_{32}	x^1_{34}	x^1_{41}	x^1_{42}	x^1_{43}	x^2_{12}	x^2_{13}	x^2_{14}	x^2_{21}	x^2_{23}	x^2_{24}	x^2_{31}	x^2_{32}	x^2_{34}	x^2_{41}	x^2_{42}	x^2_{43}	y_1	y_2	y_3	y_4
1	1	1	1	-1	0	0	-1	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
2	-1	0	0	1	1	1	0	-1	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0
3	0	-1	0	0	-1	0	1	1	1	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0
4	0	0	-1	0	0	-1	0	0	-1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	-1	0	0	-1	0	0	-1	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	1	1	1	0	-1	0	0	-1	0	0	1	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	-1	0	1	1	1	0	0	-1	0	0	1	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	-1	0	0	-1	1	1	1	0	0	0	-1
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

For constructing G_i, choose the row $i=10$. Note that G_{10} contains two components H_1 and H_2 that are not connected. Component H_1 contains set of vertices $\{1,2,3,4,9\}$ and component H_2 contains set of vertices $\{5,6,7,8,11\}$. As row $i=10$ contains non-zero coefficients in columns related to $y(T_2)$, there is no edge between H_1 and H_2.