Pickup and delivery problems with autonomous and electric vehicles

Manuel Trotta

THÈSE

Présentée par

Manuel TROTTA

pour obtenir le diplôme de

Docteur d'université

Spécialité : Informatique

Pickup and delivery problems with autonomous and electric vehicles

---

Soutenue publiquement le 24/01/2023 devant le jury

M.me ou M.	Frédéric	SEMET	Président du jury et examinateur
	Sophie	PARRAGH	Rapporteuse et examinatrice
	Manuel	IORI	Rapporteur et examinateur
	Mourad	BAÏOU	Examinateur
	Claudia	ARCHETTI	Directrice de thèse
	Dominique	FEILLET	Directeur de thèse
	Alain	QUILLIOT	Directeur de thèse

École Doctorale Sciences Pour l'Ingénieur

Université Clermont Auvergne

A nonna Maria

Acknowledgements

First and foremost I would like to express my gratitude to my PhD advisors Claudia Archetti, Dominique Feillet and Alain Quilliot. I am honored and grateful for being accepted as their PhD student. Their step-by-step guidance was illuminating when tackling the challenging problems faced throughout this thesis. In addition, I am delighted with the personal and professional relationship we have developed over the years. I would especially like to thank Dominique for the support he gave me during my first months in France and more generally during these thesis years.

Next I would like to thank my committee members Professors Manuel Iori and Sophie Parragh for the time and effort invested in the reviewing phase of this manuscript.

I also would like to thank the Laboratory of Excellence IMobS3 (Innovative Mobility: Smart and Sustainable Solutions) for funding my PhD, and the SFL (Science de la Fabrication et Logistique) department of the École des Mines de Saint-Étienne in Gardanne for hosting me and for providing all the physical conditions for the research development. The past three years at SFL have been an incredible experience, not only professionally but also personally. I thank my peer colleagues Dimitri, Karim, Lucas, Mario, Rebecca, Thibault – and many others – for their friendship and for the nice moments spent together. I would also like to thank all the other members of SFL for the good times we had together. Special thanks are due to my friend Nouaman for the nice time we have shared during my first months in France.

I thank all my family for the endless support and love, especially my parents. It is thanks to their love and efforts that I was able to start this journey. I also want to thank my brother and my sister-in-law for their love and for the good times we have spent together over these years.

Last but not least, I would like to thank my loving and supportive partner Marina for all the moments shared along these years. For all the dedication, patience, love and tenderness, thank you.

Contents

List of Figures

List of Tables

Introduction

In recent years, the development of sustainable urban mobility has become critically important. Sustainability and quality of life in cities are greatly influenced by transport. Above all, urban areas are confronted with transport-related air pollution, noise, congestion, occupation of public space by traffic, and increased morbidity and mortality rates caused by traffic accidents and pollution (Anenberg et al., 2019; Brůhová Foltýnová et al., 2020; Christodoulou & Christidis, 2020). The harmful effects of urban transport are extended by the use of fossil fuels in Internal Combustion Engines (ICEs) that contribute to global climate change; emission levels are growing rapidly and, in the absence of ambitious steps towards decarbonization, the 2016 EU Reference Scenario highlights that by 2050 road transport could account for the largest share of CO2 emissions (Capros et al., 2016). Road transport is currently the second largest source of CO2 emissions in the European Union (EU), accounting for around a quarter of total emissions (EEA, 2013).

The main promising technologies that could help to solve these problems include autonomous vehicles (AVs), electric vehicles (EVs) and the integration of shared mobility services (Miskolczi et al., 2021). Currently, shared mobility is identified as one of the most promising solutions in urban mobility in order to reduce negative externalities and to raise user satisfaction (Miskolczi et al., 2021). Other ways of reducing negative impacts of urban mobility could be a shift towards low and zero-emission modes, enhancing the role of EVs, and moderating travel demand. Reducing the burden on the environment may be achieved by minimizing travel needs and reorganizing the capacities of on-demand transportation services (Commission, 2011).

Autonomous Vehicle technology aims to reduce crashes, energy consumption, pollution, and congestion while at the same time increasing transport accessibility (Bagloee et al., 2016). In Manyika et al., 2013 vehicle automation is included on the list of the top ten disruptive technologies of the future. The forecast is that AVs are expected to constitute around 50% of vehicle sales, 30% of vehicles, and 40% of all vehicle travel by 2040 (Litman, 2017). The AV is associated with a variety of positive societal impacts such as a safer transport system, a lower cost of transport as well as enabling a modicum of mobility to the non-ambulatory and disabled as well as to those in lower income households (Bagloee et al., 2016; Hancock et al., 2019).

Such positive impacts are the driving forces behind the emergence of AV technology, making it a viable, economic model in the near future and beyond.

Electric Vehicles are a promising technology for drastically reducing the environmental burden of road transport. More than a decade ago and also more recently, they were advocated by various actors as an important element in reducing emissions of CO2, air pollutants and noise of particularly passenger cars and light commercial vehicles (van Essen & Kampman, 2011). The use of electric vehicles has heavily increased in the last years. In June 2022, the European Parliament approved a ban on the sale of all vehicles with Internal Combustion Engines (ICEs) from 2035 (ENVI, 2022). In reality, the measure approved by the Parliament aims to sell only CO2 emissions-free vehicles from 2035, but the lack of widely available alternatives means that the market for new cars will be dominated by Battery Electric Vehicles (BEVs) (McKinsey, 2022). The expansion of the electric vehicle charging network presents some critical issues. According to the McKinsey Center for Future Mobility, Europe will have to build an estimated 24 new battery giga-factories to supply EV battery demand. With more than 70 million EVs on the road by 2030, the industry will need to install a large number of public chargers and provide maintenance operations for them. Renewable electricity production needs to increase by 5% to meet EV charging demand (McKinsey, 2022).

The potential advantages that autonomous and electric vehicles offer, in relation to the urban mobility problems described above, have prompted us to investigate new Pickup and Delivery problems in which autonomous and electric vehicles are used. Pickup and delivery problems are optimization problems defined on graph structures. In this kind of problems there is a fleet of vehicles that serve transportation requests between different nodes of the graph. Each transportation request i requires transporting a load q_i from a pickup node to a delivery node. Time windows are associated with the requests. The objective of these problems is to serve all transportation requests within their respective time windows by minimizing a given objective function, such as the total distance traveled by vehicles. The main contribution of this thesis is the introduction of new Pickup and Delivery problems motivated by new mobility systems that emerge with autonomous and electric vehicles. In particular, the first part of the thesis introduces a new class of Pickup and Delivery problems defined on circular graphs. Each of the problems in this class differs in fleet composition, the presence of time windows, and the objective function to be optimized. These problems find application in the context of autonomous vehicles. Indeed, operating autonomous vehicles on separate infrastructures such as rails, guide-ways, or elevated lanes, reduces interactions with pedestrians, human-driven vehicles, and other obstacles (Kaspi et al., 2022). Such infrastructures often have special topologies such as circuits. The second part of the thesis introduces a new Pickup and Delivery problem where the fleet is composed of Battery Electric Vehicles (BEVs) which swap their used batteries at a solar-powered Battery Swapping Stations (BSS). The innovative

aspect of this problem is the optimal management of the energy used by the vehicles. In fact, the surplus energy is sold, and the objective is to serve all transportation requests by maximizing the profit made from selling the surplus energy.

In Chapter 1, an important and introductory background on the concepts exploited throughout this dissertation is provided. In Chapter 2, Pickup and Delivery problems on rings are introduced. A classification scheme is proposed, together with complexity studies and Integer Linear Programming (ILP) formulations. Chapter 3 contains the proof of NP-hardness of problem $V, 1|sd, u|CLT$ (introduced in Chapter 2) as the proof is too long. Chapter 4 focuses on Pickup and Delivery problems on rings with total completion time objective function. The complexity of all problem variants in this class is studied and an ILP formulation is proposed. In Chapter 5, Pickup and Delivery problems on rings are studied where vehicles are allowed to travel clockwise and counter-clockwise on the ring. A new algorithm is proposed for the simplest variant in this class of problems. In Chapter 6, the Electric Vehicle Pickup and Delivery Problem with Energy Management is introduced. The main features of this problem are three:

1. Electric vehicles are used to serve transportation requests;
2. The energy used to recharge electric vehicle batteries is produced in a photovoltaic power plant which is adjacent to the vehicle depot;
3. The energy can be either used to recharge the batteries or sold to the electric grid. In case the energy produced is not enough to recharge the batteries, it can also be purchased.

The problem asks for vehicle route planning and optimal energy management, with the global objective of maximizing the profit made from the sale of surplus energy.

The abstracts of the various chapters of the thesis are given below.

Abstract of Chapter 2: In this chapter a new class of Pickup and Delivery problems on circles (or rings) are introduced. These problems arise in the field of public transportation systems where autonomous (i.e. driverless) vehicles travel on circular networks. A set of stations arranged in a circle and a set of transportation requests are considered. Each request asks for the transportation of a certain quantity from a pickup station to a delivery station. A fleet of capacitated vehicles is available at the depot. In the first part of the chapter a classification scheme for these problems is proposed. In the second part, the variants in which the vehicles are allowed to move in a single direction of the circle (either clockwise or counterclockwise) are addresses, where the objective is to minimize the number of tours on the ring while serving all the requests. A complexity analysis for this class of problems is provided. Polynomial time algorithms for the variants that are polynomially solvable are proposed and

proofs of NP-hardness for the variants that are NP-hard are developed. In addition, for the latter, mathematical formulations are provided and computational tests that show the effectiveness of these formulations are performed. Finally, optimal solutions are compared with those obtained using a straightforward greedy algorithm.

Abstract of Chapter 3: This chapter contains the proof of NP-hardness of problem $V, 1|sd, u|CLT$, introduced in Chapter 2, as the proof is too long.

Abstract of Chapter 4: This chapter studies Pickup and Delivery Problems on Rings (PDP-R). PDP-R are defined on a circular network. A set of transportation requests has to be served. A fleet of vehicles is available at the depot. The objective is to assign requests to vehicles and define the service sequence for each vehicle, while minimizing the total completion time of requests. The problem is proved NP-hard. An ILP formulation is proposed and exhaustive computational tests are performed to show its effectiveness, comparing it with a straightforward greedy algorithm. Furthermore, a relax-and-repair heuristic based on a surrogate relaxation of the ILP formulation is proposed and compared with the greedy algorithm.

Abstract of Chapter 5: Pickup and Delivery problems on rings (PDP-R) are a class of problems defined on cycle graphs. This chapter is devoted to the variants in which the vehicles are allowed to travel along both directions on the ring and the objective is to minimize the time at which the last vehicle comes back to the depot with all requests served. The simplest variant of this class of problems is polynomial-time solvable and is known in the literature as the Stacker Crane Problem on circles. In this chapter, we focus on this variant. Several algorithms have been proposed in the literature. We propose a new, easier, algorithm. Other variants are left for future works.

Abstract of Chapter 6: In this chapter a problem in which a set of capacitated Battery Electric Vehicles (BEVs) carry out pickup and delivery operations with time windows constraints is studied. The energy needed to recharge the batteries of these vehicles is produced in a Battery Swapping Station (BSS) that is also the depot of the vehicles. Additional batteries are available at the depot, where vehicles can go and swap their batteries. Pickup and delivery operations must be planned over a time horizon divided into periods. In each period it must be decided how much energy to give to the batteries that are at the production unit. Also, if the energy produced is in excess of that required by the batteries, this excess can be sold to the general network at a profit. If the energy required by the batteries is greater than the energy produced, an unlimited amount of energy can be bought from the general network. The objective of the problem is to plan vehicle routes to meet all pickup and delivery demands while maximizing the net profit that is made from the energy sold and bought over the time horizon. An MILP formulation of the problem is is proposed and a matheuristic approach is developed. The matheuristic approach consists of three steps: in the first one a subset of feasible trips is generated by using a

Randomized Construction Heuristic, in the second step the formulation is solved over this set of trips, and in the third one a repair procedure is performed on the obtained solution, in order to avoid more than one trip visiting the same node. Computational tests on modified Li and Lim's benchmark instances for the PDPTW are performed and the impact of the parameters on the hardness of these instances is studied.

The work done in the first part of the thesis resulted in the publication of the paper Trotta et al., 2022 and in the technical paper Trotta et al., 2021. Some results of the first part of the thesis were also presented at conferences *Odysseus 2021, the Eight International Workshop on Freight Transportation and Logistics, May 2022, Tangier, Morocco* and *21st annual congress of the Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF 2020), University of Montpellier, Feb 2020, Montpellier, France.*

The work done in the second part of the thesis was presented at conferences *11th Triennial Symposium on Transportation Analysis conference (TRISTAN XI) June 19-25, 2022, Mauritius Island* and *23rd annual congress of the Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF 2022), INSA Lyon, Feb 2022, Villeurbanne - Lyon, France.*

Chapter 1

Preliminaries and notation

This chapter introduces some preliminary definitions, notations and basics that will make reading this manuscript easier.

1.1 Graph Theory

The content of this section is based on Berge, 1985, Murty and Bondy, 2008 and Wahlström, 2018.

Graphs, vertices and edges. A *graph* is an ordered pair $G = (V, E)$, where V is a non-empty set of objects whose elements are called *vertices*, and E is a multiset of unordered pairs of (not necessarily distinct) vertices, called *edges*. Throughout this manuscript V and E are finite sets. An edge of the form (u, u) is called a *loop*.

Incidence, adjacency and degree. If $e = (u, v)$ is an edge then e is said to join u and v, and the vertices u and v are called the *endpoints* of e. If the direction of the edge $e = (u, v)$ is specified, then u is called the *initial endpoint* and v is called the *terminal endpoint*, and e is called *arc*. The endpoints of an edge are said to be incident with the edge, and *vice versa*. Two vertices which are incident with a common edge are *adjacent*, as are two edges which are incident with a common vertex. If a vertex u is the initial endpoint of an arc $e = (u, v)$ which is not a loop, the arc e is said to be *incident out of vertex* u. The number of arcs in graph G that are incident out of u plus the number of loops attached to u is denoted by $d^+_G(u)$ and is called the outer degree of u. The inner degree of a vertex v is denoted by $d^-_G(v)$ and it is defined similarly. The *degree* of a vertex $v \in V$ is the number of arcs with v as endpoint, each loop being counted twice. It is denoted by $d_G(v) = d^+_G(u) + d^-_G(v)$.

Directed graphs, undirected graphs and simple graphs. If set E is composed of unordered pairs of vertices, the graph is said to be *undirected* or a *multigraph*.

If instead E is composed of ordered pairs of vertices, the graph is said to be *directed*. A multigraph is *simple* if it has no loops or parallel edges. Graphs with loops are simply called graphs if it is clear from context.

Paths, cycles and circuits. A *path* $p = (u_1, \ldots, u_q)$ is a sequence of arcs of G such that the terminal endpoint of arc u_i is the initial endpoint of arc u_{i+1} for all $i < q$. The *length* of a path is the number of arcs in the sequence. A path that does not encounter the same vertex twice is called *elementary*. A path that does not use the same arc twice is called *simple*. A *cycle* or *circuit* is a simple path $c = (u_1, \ldots, u_q)$ where the initial endpoint of the path corresponds to its final endpoint [1].

Acyclic graphs. A graph is *acyclic* if it does not contain any cycle. A directed acyclic graph admits at least an ordering $v_1, v_2, \ldots, v_n$ of its vertices such that for every directed arc uv from vertex u to vertex v, u comes before v in the ordering. This ordering is called *acyclic ordering* or *topological ordering*.

Subgraphs. The subgraph of G generated by $X \subset V$ is the graph with X as its vertex set and with all the arcs in G that have both endpoints in X.

Connectivity. *A connected graph* is a graph that contains a path between each pair (u, v) of distinct vertices. A *connected component* is a maximally connected subgraph of G. In other words, if $X \subset V$ is the vertex set of the subgraph, for every vertex $v \in V \setminus X$, the subgraph generated by $X \cup \{v\}$ is not connected. Connected components in directed graphs are called *strongly connected components*.

Eulerian and semi-Eulerian graphs. An *Eulerian cycle* in graph G is a cycle that uses each edge of the graph exactly once. A graph is said to be *Eulerian* if and only if it contains an Eulerian cycle. If a graph contains a path from a vertex u to a vertex v that uses each arc exactly once, then it is said to be *semi-Eulerian*. In Wahlström, 2018, semi-Eulerian directed graphs are called *balanced*.

Vertex coloring. A *vertex-coloring* of a graph is a function $f : V \to \mathbb{N}$ that assigns natural numbers (or colors) to its vertices and is such that if $f(u) \neq f(v)$ for every $u \neq v \in V$. If a graph admits a coloring it is said to be *colorable*. A *k-coloring* of a graph is a coloring that uses exactly k colors.

Intersection graphs, interval graphs and circular-arc graphs. A *set system* is an ordered pair $(V, \mathcal{F})$ where V is a set of elements and $\mathcal{F}$ is a family of subsets of V. With each set system $(V, \mathcal{F})$ one may associate its *intersection graph*. This is the graph whose vertex set is $\mathcal{F}$, two sets in $\mathcal{F}$ being adjacent if their intersection is non-empty. When $V = \mathbb{R}$ and $\mathcal{F}$ is a set of closed intervals of $\mathbb{R}$, the intersection graph of $(V, \mathcal{F})$ is called an *interval graph*. A graph is called a *circular-arc graph* if it is the intersection graph of a family of arcs on a circle. Circular-arc graphs generalize interval graphs.

[1] Berge, 1985 defines cycles and circuits slightly differently, but in this manuscript they are used as synonyms.

1.2 Algorithms and Computational Complexity Theory

The content of this section is based on Ausiello et al., 1999 and Arora and Barak, 2009.

Computational problems. A computational problem can be expressed in terms of some *relation* $P \subseteq I \times S$, where I is the set of *problem instances* and S is the set of *problem solutions*. If $(x, y) \in P$ we say that y is a solution for instance x.

Decision problems, search problems and optimization problems. A *decision problem* asks to determine if an instance x satisfies a given condition. In this case relation P reduces to a function $f : I \to S$ where S is the set $\{yes, no\}$. A *search problem* asks to find, for any instance $x \in I$, a solution $y \in S$ such that $(x, y) \in P$ is verified. An *optimization problem*, given an instance $x \in I$, consists in finding the best solution y^* - according to some measure - among all solutions $y \in S$ such that $(x, y) \in P$ is verified.

Computational models and cost measures. In computational complexity theory, a computational model is generally chosen as a reference. In this thesis, no assumption is made about the computational model since it seems that all physically realizable computational models can be simulated by a single abstract computational model such as *Turing machines* Arora and Barak, 2009. However, to measure the time it takes an algorithm to calculate the solution to a problem (known as *running time*), the *uniform cost measure* will be employed, which consists of determining the total number of elementary instructions executed by an algorithm before halting. It assumes that any elementary operation can be executed in constant time on operands of any size.

Asymptotic complexity and worst case analysis. In this thesis, measures of algorithm efficiency (such as running time) do not specify the exact time it takes an algorithm to compute the solution to a given instance of a problem, but rather the behavior of running time as a function of instance size. That is, the efficiency of an algorithm can be measured by a function f from the set $\mathbb{N}$ of natural numbers to itself such that $f(n)$ is equal to the maximum number of basic operations that the algorithm performs on inputs of length n. This kind of behavior of an algorithm is known as *asymptotic complexity*. However, instances of the same size can have very different running times. Therefore, the cost of an algorithm is defined as the running time of the algorithm on the *worst case instance*, i.e., the instance that causes the highest running time. The *instance size* (or *input size*) is assumed to be the number of digits needed to represent the instance of the problem.

The big-O notation. If f, g are two functions from $\mathbb{N}$ to $\mathbb{N}$, then we say that $f = O(g)$ if there exists a constant c and a natural number n_0 such that $f(n) \leq cg(n)$

for all $n \geq n_0$. If the number of basic operations of an algorithm is $3n^2 + 4\log(n)$ on an instance of size n, we say that its running time is $O(n^2)$.

Upper bound on the complexity of a problem. Given a problem P we say that the *time complexity upper bound of* P is $O(f(n))$ if there exists an algorithm that solves all instances of P and whose running time is $O(f(n))$.

P and NP complexity classes. A *complexity class* is a set of problems that can be solved withing given resource bounds. For any function $f(n)$, let TIME($f(n)$) be the collection of decision problems that have time complexity $O(f(n)$. The complexity class P is the class of all problems solvable in time proportional to a polynomial of the input size. Formally, $P = \bigcup_{k=0}^{\infty} \text{TIME}(n^k)$. A *nondeterministic algorithm* is an algorithm that can execute instructions of the type "guess $y \in \{0, 1\}$". Essentially, a nondeterministic algorithm has the additional option of "guessing" a value. For any function $f(n)$, let NTIME($f(n)$) be the collection of decision problems that can be solved by a nondeterministic algorithm in time complexity $O(f(n)$. The complexity class NP is the class of all decision problems that can be solved in time proportional to a polynomial of the input size by a nondeterministic algorithm. Formally, $NP = \bigcup_{k=0}^{\infty} \text{NTIME}(n^k)$.

Karp-reducibility and polynomial time reductions. A problem P is a *decision problem* if the set I_P of all instances of P can be partitioned into a set Y_P of *positive instances* and a set N_P of *negative instances* and the problems asks, for any instance $x \in I_P$, to verify if $x \in Y_P$. A decision problem P_1 is said to be *Karp-reducible* to a decision problem P_2 if there exists an algorithm R that, given any instance x of P_1, transforms it into an instance y of P_2 in such a way that $x \in Y_{P_1}$ if and only if $y \in Y_{P_2}$. In such a case, R is said to be a Karp-reduction from P_1 to P_2. As a consequence, if a decision problem P_1 is Karp-reducible to a decision problem P_2, then for any instance x of P_1, x is a positive instance if and only if the transformed instance y is a positive instance for P_2. A special case of Karp-reducibility is the polynomial-time Karp-reducibility. A decision problem P_1 is polynomially Karp-reducible to a decision problem P_2 if and only if it is Karp-reducible and the corresponding reduction R has polynomial-time complexity. In this manuscript, all reductions used in the complexity analyses of the problems are polynomial-time Karp-reductions.

Hardness and completeness of decision problems. For any complexity class C, a decision problem P is said to be *C-hard* with respect to a reducibility $\leq_r$ if, for any decision problem $P' \in C$, $P' \leq_r P$. If P is C-hard and $P \in C$, P is said to be $C - complete$. A decision problem P is said to be *NP-hard* if it is hard in NP with respect to the polynomial-time Karp-reducibility, that is, for any decision problem $P' \in NP$, $P' \leq_m^p P$. A decision problem P is said to be NP-complete if it is NP-hard and $P \in NP$. By the definition of NP-hardness and by the transitive property of polynomial-time Karp-reductions, any decision problem P can be proved to be

NP-hard by providing a polynomial-time Karp-reduction from some other problem P' - already known to be NP-hard - to P. This technique is used in almost all proofs of NP-hardness in this thesis.

Complexity of optimization problems. Any optimization problem P has an *associated decision problem* P_D. In the case that P is a minimization problem, P_D asks, given an instance x and a positive integer K, for the existence of a feasible solution y with value $\leq K$. Similarly, if P is a maximization problem, the associated decision problem asks, given a positive integer K, for the existence of a feasible solution y with value $\geq K$. For any optimization problem P, the corresponding decision problem P_D is not harder than P. In fact, to solve P_D on an instance x it is sufficient to run some algorithm for P to obtain the optimal solution y^* and its optimal value z^*; then, it is sufficient to check if $z^* \leq K$ in the minimization case ($z^* \geq K$ in the maximization case). For this reason, almost all proofs of NP-hardness in this thesis use decision versions of optimization problems.

1.3 Linear and Integer Programming

Let $\mathbb{R}^n$ denote the set of real-valued vectors of dimension n. A *linear program* (LP) consists in finding a vector $x \in \mathbb{R}^n$ that minimizes (or maximizes) a linear function $f(x)$ over a finite set of linear inequalities $ax \leq b$, where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. Therefore, a linear optimization problem can be stated as

$$\min\{f(x) = c^T x \text{ such that } Ax \leq b\}$$

where c^T is an n-dimensional real-valued row vector. A vector $x \in \mathbb{R}^n$ is a *feasible solution* for the system of linear inequalities $Ax \leq b$ if it satisfies all inequalities in $Ax \leq b$. If the system of inequalities does not admit any feasible solution, the problem is said to be *infeasible*. The linear function $c^T x$ is called the *objective function*. A feasible solution x that minimizes (or maximizes) the objective function is called an *optimal solution*. If a problem has a feasible solution but does not have an optimal solution, it is said to be *unbounded*. An integer linear program (ILP) is a linear program with the additional restriction that vector x can assume only integer values. The formulation of an ILP can thus be stated as

$$\min\{f(x) = c^T x \text{ such that } Ax \leq b, x \in \mathbb{Z}^n\}$$

If only some decision variables are constrained to assume integer values, the problem is called a *mixed-integer linear program* (MILP) and can be stated as

$$\min\{f(x, y) = c^T x + h^T y \text{ such that } Ax + Gy \leq b, x \in \mathbb{Z}^n, y \in \mathbb{R}^p\}$$

Given an ILP $max\{f(x) = c^T x \text{ such that } Ax \leq b, x \in \mathbb{Z}^n\}$, its *linear relaxation* is the LP $\max\{f(x) = c^T x \text{ such that } Ax \leq b, x \in \mathbb{R}^n\}$ obtained by removing the integrality constraints.

Chapter 2

Pickup and Delivery problems with autonomous vehicles on rings

This chapter is based on the article:

- Trotta, M., Archetti, C., Feillet, D. and Quilliot, A. (2022). "Pickup and delivery problems with autonomous vehicles on rings". In: *European Journal of Operational Research 300*(1), pp. 221–236.

In this chapter a new class of Pickup and Delivery problems on circles (or rings) is introduced. These problems arise in the field of public transportation systems where autonomous (*i.e.* driverless) vehicles travel on circular networks. A set of stations arranged in a circle and a set of transportation requests are considered. Each request asks for the transportation of a certain quantity from a pickup station to a delivery station. A fleet of capacitated vehicles is available at the depot. In the first part of the chapter a classification scheme for these problems is proposed. In the second part, the variants in which the vehicles are allowed to move in a single direction of the circle (either clockwise or counterclockwise) are addresses, where the objective is to minimize the number of tours on the ring while serving all the requests. A complexity analysis for this class of problems is provided. Polynomial time algorithms for the variants that are polynomially solvable are proposed and proofs of NP-hardness for the variants that are NP-hard are developed. In addition, for the latter, mathematical formulations are provided and computational tests that show the effectiveness of these formulations are performed. Finally, optimal solutions are compared with those obtained using a straightforward greedy algorithm.

2.1 Introduction

It is a matter of fact that urban mobility, by means of public transport or personal cars, has become a key factor in people's everyday life. More and more people commute on urban areas on a daily basis. This clearly causes a number of issues like traffic congestion and concentration of polluting emissions, and public entities are struggling to find ways and policies that help in facing the ever growing demand of mobility services. For an overview of the main challenges transport is facing both at the EU-level and in Member States, see "Transport in the European Union: Current Trends and Issues", 2019. Among the different solutions proposed in recent years, this chapter focuses on autonomous vehicles like cars, minibuses and shuttles. The use of these vehicles raises legal, ethical, economic and safety issues. Due to these problems it is not likely that autonomous vehicles will totally replace normal vehicles soon. However, they will probably first be authorized for collective transportation.

There already exist a few cases where fully autonomous (i.e. driverless) vehicles are used in public transportation. In May 2019 Groupe Renault, the Transdev Group, IRT SystemX, Institut VEDECOM and the University of Paris-Saclay initiated a new project called Paris-Saclay Autonomous Lab ("Paris-Saclay Autonomous Lab: new autonomous, electric and shared mobility services", 2019). Its purpose was to develop new autonomous mobility services using dedicated lanes and public streets to supplement the existing Saclay Plateau transportation system. An overnight public transportation service using an autonomous Transdev-Lohr i-Cristal shuttle was designed to serve the Saclay Plateau neighborhoods from the Massy station. On December 2018, Keolis and the European Metropolis of Lille launched an electric autonomous shuttle service at the University of Lille in Villeneuve d'Ascq, with a population of 20,000 students and 1,600 researchers ("Keolis deploys electric autonomous shuttles at two university campuses in France", 2018). The service at Lille university had employed two Navya electric autonomous shuttles for a period of one year with four dedicated stops on a 1.4 km circle route, and had provided connections to two metro stations. More recently, in Lyon a project has been launched to study the use of electric autonomous shuttles for urban mobility ("Projet AVENUE : Navettes autonomes en milieu urbain", 2020).

In Antonialli, 2019 a worldwide benchmark on the use of Autonomous Shuttles for Collective Transport (ASCT) has been performed. By the time this research was carried out, a total of 92 experiments were identified, spread over 32 countries around the world and enabled by 20 different autonomous shuttle manufacturers. Results showed a European lead on both the number of experiments and manufacturers, with highlights to the French startups Navya and EasyMile. Regarding the road environment, two distinct scenarios were observed. In the first, shuttles circulate in closed/controlled areas (such as university campuses, parks, hospitals, resorts, airports, and other designated roads); this kind of deployment accounted for 52.17%

of the projects. In the second scenario (47.83%), shuttles were able to circulate among mixed traffic – for these cases the routes were mainly predetermined in city-centers or areas with a slow-speed circulation for regular vehicles. By analyzing the prevailing business models, the author observed that the vast majority of experiments tackled public transport schemes (96.55%) with daily commuters as the main revenue source for the transport operator. Systems with regular lines accounted for the vast majority of models among the sampled projects (91.21%) while demand-responsive transport answered to only 4.40% and a mixed approach comprising both operation models was present in the other 4.40%. Notwithstanding, as more countries and cities begin to allow testing and circulation of autonomous vehicles, the percentage of on-demand autonomous mobility is likely to increase.

In this work, on-demand transportation services carried out using autonomous vehicles on circular networks are investigated. Circular networks are typical in the closed/controlled areas mentioned above. These problems belong to the class of Vehicle Routing Problems with Pickups and Deliveries (VRPPD). Pickup and delivery problems are a class of vehicle routing problems in which objects or people have to be transported from origins to destinations while minimizing a given objective function. Usually this kind of problems are modelled using directed graphs where nodes represent locations/customers and arcs represent links among them. The aim of this chapter is to introduce and study a new class of pickup and delivery problems where the underlying graph is a circle. In particular, a setting where a set of stations are arranged in a circle and a set of transportation requests have to be satisfied is considered. Each request asks for the transportation of a certain quantity from a pickup station to a delivery station. A fleet of capacitated vehicles is available at the depot. The objective is to serve all transportation requests while optimizing a given objective function.

For an overview on the VRPPD the reader is referred to Desaulniers et al., 2002. For a survey on pickup and delivery problems the reader is referred to Parragh et al., 2008a, 2008b, where the problems are classified as Pickup and Delivery Vehicle Routing Problem (PDVRP), where pickup and delivery points are unpaired, the Pickup and Delivery Problem (PDP), where pickup and delivery points are paired, and the Dial-A-Ride Problem (DARP) which deals with passenger transportation between paired pickup and delivery points. The research on Pickup and Delivery problems is still very active (Wu et al., 2019), (Rüther & Rieck, 2020). The reader is referred to Berbeglia et al., 2007; Berbeglia et al., 2010 for surveys on static and dynamic Pickup and Delivery problems. Toth et al., 2014 contains two chapters on the PDP, respectively the PDP for goods transportation (Battarra et al., 2014) and for people transportation (Doerner & Salazar-González, 2014). For a review article on the DARP the reader is referred to Cordeau and Laporte, 2007. For more recent surveys on dial-a-ride problems, see Ho et al., 2018 and Molenbruch et al., 2017. For a high-level classification of dial-a-ride problems, the reader is referred to Gökay

et al., 2019.

Concerning problems defined on circles, the existing literature is limited. In Guan, 1998 the multiple capacity nonpreemptive vehicle routing problem on cycles is studied. Gendreau et al., 1999 developed a linear time exact algorithm for the Single-Vehicle Pickup and Delivery Problem defined on a cycle graph. Tzoreff et al., 2002 studied the Vehicle Routing Problem with Pickup and Delivery on some special graphs. They developed an optimal algorithm that runs in polynomial time for cycle graphs.

Ilani et al., 2015 presented some optimal polynomial-time algorithms for two variants of the Fixed Route DARP with a circular route. The first one considers a fleet of infinite capacity vehicles, while the second one considers the more general case of vehicles with heterogeneous capacities. Dial-a-Ride problems with autonomous vehicles have also been studied in Pimenta et al., 2017a and Baïou et al., 2018.

A number of related problems arise in the field of industrial automation. Atallah and Kosaraju, 1988 were probably the first to study the problem of efficiently rearranging parts in the plane with a centrally placed gripper that can rotate. This problem is known as the Stacker Crane Problem (SCP), and they proposed a polynomial time algorithm for the SCP on a circle. Anily and Pfeffer, 2013 studied a similar problem, the Uncapacitated Swapping Problem, on a line and on a circle, where the objective is to rearrange objects of different types on a circular graph using an uncapacitated vehicle. It can be seen as a generalization of the SCP. They proposed a polynomial time algorithm for both cases of a line and a circle.

The contributions of this chapter can be summarized as follows. A new class of problems, the Pickup and Delivery Problems on Rings (PDP-R), is introduced. A classification scheme for these problems is proposed which resembles the classification used for scheduling problems. In this class, a subclass where vehicles all travel in the same direction along the circle is investigated, where the objective is to minimize the time at which the last vehicle returns to the depot. The peculiarity of these problems is that they do not involve any routing decision: the vehicles repeatedly turn around the circle until all pickup and delivery services have been carried out. The optimization comes from assigning delivery services to vehicles in such a way that a given objective function is optimized. The computational complexity for all variants of this subclass is determined. Each variant is obtained with a different combination of the following parameters: number of vehicles, vehicle capacity, direction of movement along the ring, presence of release and due dates, objective function. Polynomial time algorithms for the problems that are polynomially solvable are developed and proofs of NP-hardness for the others are proposed. In addition, for the latter, efficient mathematical formulations are provided that allow solving large-size instances quickly. Finally, optimal solutions are compared with those obtained using a straightforward greedy algorithm, that could be easily implemented by practition-

ers.

The rest of the chapter is organized as follows. In Section 2.2, the notation is introduced together with some basic definitions, and the classification scheme is presented. Section 2.3 is dedicated to problems with unitary requests and no release/due dates. Section 2.4 is devoted to problems with unitary requests and release/due dates. Section 2.5 deals with problems having non-unitary requests. Computational experiments and the greedy algorithm are presented in Section 2.6. In Section 2.7, the results are summarized and some perspectives of this work are presented.

2.2 Problem description, classification and scope of the chapter

In this section, a general description of the PDP-R is first provided. Then, a classification scheme is introduced. Finally, the variants addressed in this chapter are presented.

2.2.1 General problem description

The setting of the PDP-R is the following. The ring is represented by a directed graph whose set of nodes includes m stations numbered from 0 to $m-1$. To simplify further notation, station 0 is indifferently denoted as 0 or m. The set of arcs consists of $2 \times m$ links $(j, j+1)$ and $(j+1, j)$ between consecutive stations $(0 \leq j \leq m-1)$ (see Figure 2.1). Travel times $\delta_{j,j+1}$ and $\delta_{j+1,j}$ are defined between two consecutive stations $(j = 0, ..., m-1)$. These travel times are not necessarily symmetric.

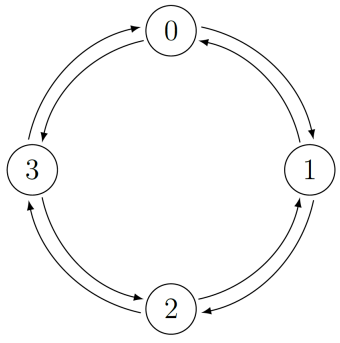


Figure 2.1: PDP-R with four stations (station 0 is the depot)

A multiset $\mathcal{R}$ of n transportation requests is considered. In the more general case, each request $i \in \mathcal{R}$ is defined by: a pickup station s_i and a target station

t_i ($s_i, t_i = 0, ..., m-1,\ s_i \neq t_i$), a quantity q_i to be transported, a release date r_i and a due date d_i. The release date indicates the earliest time for the pickup operation, the due date is the latest time for the delivery. $\mathcal{R}$ is defined as a multiset instead of a set because it can contain identical requests.

The requests are served by a fleet of vehicles that can travel along the ring in either one or both directions. The vehicles are capacitated, so the load they can transport at the same time cannot exceed their capacity. Station 0 is the depot, *i.e.,* a station where the vehicles are located before starting the service and have to return after having served all the requests. Without loss of generality, travel times are assumed to be equal to distances, and therefore the words time and distance are interchangeable.

2.2.2 A classification scheme

One of the objectives of this work is to propose a classification scheme for PDP-R problems. It has been inspired by the three-field classification introduced in Graham et al., 1979 for scheduling problems. It is composed of three fields - α, β and γ - separated by a vertical bar. Each field may be a comma separated list of words that describes one or more features of the problem.

The α field contains information on the vehicles, separated by a comma:

- number of available vehicles: 1 or V (with $V > 1$);
- capacity of the vehicles: 1 or Q (with $Q > 1$) or Q_v (if vehicles have different capacities).

The β field reports the direction of movement (mandatory) and 3 optional constraints. The direction of movement can have 3 different values:

- *sd* (single direction): vehicles all follow the same direction, fixed from the beginning (clockwise or counterclockwise);
- *md* (mixed single-direction): each vehicle follows a single direction, fixed from the beginning, that can be either clockwise or counterclockwise;
- *bd* (both directions): each vehicle can follow both directions.

The optional constraints, separated by a comma, are:

- r_i: pickups are subject to release dates;
- d_i: deliveries are subject to due dates;

- u: demands are unitary.

Finally, field γ indicates the objective function:

- C_{max}: makespan, *i.e.*, the completion time of the last request scheduled among all the vehicles;
- $\sum C_i$: total completion time, i.e., the sum of completion times over all vehicles;
- CLT: closing time, *i.e.*, time at which the last vehicle comes back to the depot when all requests are served.

For example, the PDP-R with one vehicle of unit capacity rotating clockwise with no release dates and due dates whose objective is the minimization of the maximum completion time is denoted by $1, 1|sd, u|C_{max}$.

Among the papers cited in Section 2.1, two of them describe problems defined on cycle graphs that belong to the classification scheme proposed in this chapter. The first is Guan, 1998 and concerns pickup and delivery problems on paths, cycles and trees. One of the problems described in this chapter is the multiple capacity nonpreemptive vehicle routing problem on cycles that is shown to be NP-complete. The corresponding problem in our classification is $1, Q|bd|CLT$. The second is the Stacker Crane Problem (SCP) on a circle described in Atallah and Kosaraju, 1988. The corresponding problem in our classification scheme is $1, 1|bd, u|CLT$.

As for the other problems defined on cycle graphs, they do not belong to our classification scheme for the reasons developed below:

- Gendreau et al., 1999: the closest problem in our classification is $1, Q|bd|CLT$ where stations represent customers. However, there are some important differences. In the former:
  - in any feasible solution the vehicle leaves the depot with a load equal to the sum of delivery demands and gets back to the depot with a load equal to the sum of pickup demands
  - if a customer requires both pickup and delivery, the two operations must be serviced at the same time
  - the sum of pickup demands, as well as the sum of delivery demands, is smaller than Q

  The consequence is that any feasible solution has an optimal value not greater than $2L$ (with L the length of the ring).

- Tzoreff et al., 2002: in this paper the authors study the same problem studied in Gendreau et al., 1999 but with possibly two depots on some special graphs, including cycle graphs. They also make similar hypotheses.
- Ilani et al., 2015: in this paper the problem aims at minimizing the average waiting time for customers in a mono-directional setting with a mixed fleet of vehicles. Vehicles are therefore allowed to wait, and it makes a big difference The objective function and the fact that vehicles are allowed to wait are not considered in our classification.
- Pimenta et al., 2017b and Baïou et al., 2018: in these papers the authors study the Stop Number Minimization Problem (SNMP), where the objective is to minimize the number of stops of the vehicles in a mono-directional setting. Since the objective function is different from the ones taken into consideration, the SNMP does not belong to the classification scheme introduced above.

2.2.3 Scope of the chapter and related definitions

This chapter contains theoretical results and computational experiments concerning all the problem variants where the vehicles are allowed to move on the ring in a single direction (sd) and the objective function is the minimization of the closing time (CLT). Without loss of generality, all vehicles are assumed to move clockwise.

Due to the circular layout, to go from j_1 to j_2, vehicles must pass through all intermediate stations between j_1 and j_2, that is: stations $j_1+1,\dots,j_2-1$ if $j_1 < j_2$, stations $j_1+1,\dots,m-1,0,\dots,j_2-1$ otherwise. This allows us to define the distance between two stations from distances between consecutive stations:

$$\delta_{j_1,j_2} = \begin{cases} \sum_{k=j_1}^{j_2-1} \delta_{k,k+1} & \text{if } j_1 < j_2 \\ \sum_{k=j_1}^{m-1} \delta_{k,k+1} + \sum_{k=0}^{j_2-1} \delta_{k,k+1} & \text{otherwise} \end{cases} \tag{2.1}$$

Note that distances δ_{j_1,j_2} satisfy the triangle inequality. We call L the length of a complete tour, starting from the depot and returning back to the depot.

To ease the readability in the remainder of the chapter, a few definitions are introduced.

Definition 1. *We say that a request $i \in \mathcal{R}$ covers a segment $[j, j+1]$ if stations j and $j+1$ are within stations s_i and t_i when going clockwise. Equivalently, it means that a vehicle needs to traverse arc $(j, j+1)$ in order to serve i.*

Definition 2. *We say that a request $i \in \mathcal{R}$ covers a station j if it covers both segments $[j-1, j]$ and $[j, j+1]$.*

In particular, a request i *covers* the depot when $0 < t_i < s_i$.

Definition 3. *We say that two requests $i_1, i_2 \in \mathcal{R}$ overlap if at least one segment $[j, j+1]$ of the ring is covered by both requests $(j = 0, ..., m-1)$.*

Definition 4. *If two requests do not overlap they are said to be compatible. Otherwise, they are said to be in conflict (or conflicting).*

Figure 2.2 shows an example with four stations and three requests identified by their pair (s_i, t_i): $(3, 1)$, $(1, 2)$, $(1, 0)$. Requests $(3, 1)$ and $(1, 2)$ are compatible, while requests $\{(1, 2), (1, 0)\}$ or $\{(1, 0), (3, 1)\}$ are in conflict (requests overlap in both pairs).

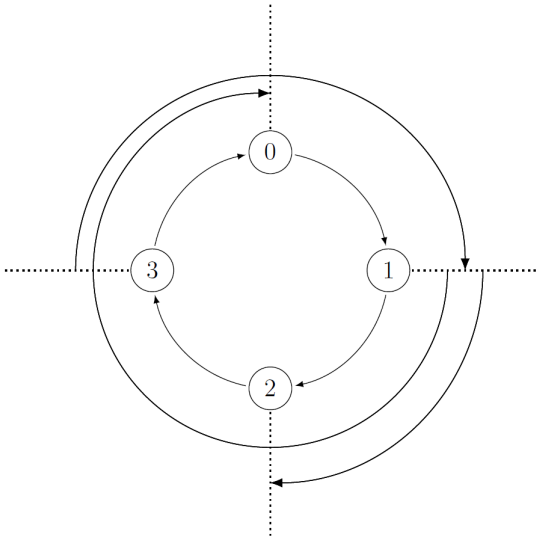


Figure 2.2: An example of compatible and overlapping requests

Definition 5. *A tour is a complete rotation of a vehicle around the ring, starting from station 0, passing through all stations j $(j = 1, ..., m-1)$ and getting back to 0.*

Definition 6. *A schedule is an assignment of the requests to the vehicles that specifies, for each vehicle, the tour in which the requests are executed. A schedule is also a solution for the problem.*

Definition 7. *A request is said active at a given time when the service of this request is started but not finished, i.e., the request has been picked up at station s_i but not yet delivered to station t_i.*

2.3 Problems with unitary requests and no release/-due dates

In this section, problem $1,1|sd,u|CLT$ is proven to be polynomially solvable. In this case, the fleet is limited to a single vehicle of unit capacity and requests have unitary demands. No release dates nor due dates are considered.

Then, other classes of problems with unitary demands and no release/due dates are investigated. Remember that, as all other problems addressed in this chapter, vehicles are subject to the *sd* constraint (single-direction) and that the objective is CLT (closing time). These problems are proven NP-hard (Section 2.3.2) and an Integer Linear Programming formulation is proposed (Section 2.3.3).

2.3.1 Problem $1,1|sd,u|CLT$

A single vehicle of capacity one has to serve the multiset $\mathcal{R}$ of n unitary requests. It is constrained to travel clockwise. The objective is to serve all requests and return to the depot as early as possible. Equivalently, it consists in minimizing the number of tours traveled by the vehicle to serve all requests.

Note that if there is no request that covers the depot, the problem reduces to the coloring of an interval graph, where requests represent intervals and tour numbers represent colors. This problem is known to be polynomial. With requests that cover the depot, this structure is lost: colors cannot be matched to tour numbers anymore. For example a simple instance with a single request covering the depot requires two tours and a single color.

A polynomial-time algorithm based on a completely different idea is proposed. The principle is to construct a graph whose vertices represent the stations and whose arcs represent the requests, and to transform this graph into an Eulerian graph by adding dummy requests that do not change the value of the optimal solution. The Eulerian tour then provides the order in which the requests should optimally be satisfied.

The following notation is used. Let T_{OPT} denote the optimal number of tours. Let $z(\mathcal{R})$ denote the optimal closing time. z is expressed as a function of $\mathcal{R}$ because dummy requests are introduced in the algorithm, as explained later. Note that $z(\mathcal{R}) = L \times T_{OPT}$. Requests are defined by their origin and their destination; each is represented by the pair (s_i, t_i). Then, $\mathcal{R} = \{(s_i, t_i) : 1 \leq i \leq n\}$. For all stations j $(j = 0, ..., m-1)$:

- $N(j, j+1)$ is the number of requests that cover segment $[j, j+1]$;
- $N^*(j)$ is the number of requests that cover station j;

- $N^{sup} = \max(N(j, j+1) : 0 \leq j \leq m-1)$. N^{sup} is the maximum number of pairwise overlapping requests.

Note that $N^*(j) \leq N(j, j+1)$ and, thus, also, $N^*(j) \leq N^{sup}$ for $j = 0, ..., m-1$.

The pseudo-code of the algorithm for solving problem $1, 1|sd, u|CLT$ is provided in Algorithm 1.

Algorithm 1 Solution algorithm for problem $1, 1|sd, u|CLT$

1: **for all** $j \in \{0, \ldots, m-1\}$ **do**
2: compute $N(j, j+1)$ and $N^*(j)$
3: **end for**
4: compute N^{sup}
5: $\mathcal{R}^{(1)} \leftarrow \mathcal{R}$
6: **for all** $j \in \{0, \ldots, m-1\}$ **do**
7: add $N^{sup} - N(j, j+1)$ copies of $(j, j+1)$ to $\mathcal{R}^{(1)}$
8: **end for**
9: $\mathcal{R}^{(2)} \leftarrow \mathcal{R}^{(1)}$
10: **if** $N^*(0) = N^{sup}$ **then**
11: **for all** $j \in \{0, \ldots, m-1\}$ **do**
12: add $(j, j+1)$ to $\mathcal{R}^{(2)}$
13: **end for**
14: **end if**
15: $\mathcal{G} \leftarrow (\{0, \ldots, m-1\}, \mathcal{R}^{(2)})$
16: $\{(C_1, n_1^{inf}), ..., (C_P, n_P^{inf})\} \leftarrow decomposeEulerian(\mathcal{G})$
17: $\Lambda \leftarrow C_1$
18: **if** $P > 1$ **then**
19: **for all** $p \in \{2, \ldots, P\}$ **do**
20: add $(n_{p-1}^{inf}, n_p^{inf})$ to Λ
21: add C_p to Λ
22: **end for**
23: add (n_P^{inf}, n_1^{inf}) to Λ
24: **end if**
25: **return** Λ

In Lines 1 to 4, values $N(j, j+1)$ and $N^*(j)$ for $j = 0, \ldots, m-1$ are computed. Then, N^{sup} is computed.

Lines 5 to 8 complete the multiset of requests so that the number of requests covering every segment $[j, j+1]$ is exactly N^{sup}. To this end, artificial requests $(j, j+1)$ are added. Let $\mathcal{R}^{(1)}$ denote the new multiset. We will see in Lemma 2 that adding these requests does not change the optimal solution value: $z(\mathcal{R}^{(1)}) = z(\mathcal{R})$. If the number of requests covering station 0 is equal to N^{sup}, the request multiset

is modified again in Lines 9 to 14. The new multiset is called $\mathcal{R}^{(2)}$. In addition to $\mathcal{R}^{(1)}$, it contains a request $(j, j+1)$ for every segment $[j, j+1]$. Again, we will see in Lemma 4 that $z(\mathcal{R}^{(2)}) = z(\mathcal{R}^{(1)})$.

The next step consists in introducing a directed multigraph $\mathcal{G}$ (Line 15). For sake of brevity $\mathcal{G}$ will be referred to as a graph, although it is always a multigraph. In this graph, the vertex set is the set of stations. An arc is added between two stations j_1 and j_2 for each request (j_1, j_2) in the extended multiset of requests. A cost δ_{j_1,j_2} is defined for each arc (j_1, j_2). Lemma 5 will prove that graph $\mathcal{G}$ is semi-Eulerian (i.e. for every node x, the in-degree of x is equal to its out-degree).

Semi-Eulerian graphs can be decomposed into a set of edge-disjoint and vertex-disjoint Eulerian circuits. Procedure *decomposeEulerian*($\mathcal{G}$) at Line 16 executes this decomposition. It results in a set of P circuits C_1 to C_P. In addition, it computes station n_p^{inf} of minimal index in every circuit C_p. Without loss of generality, it is assumed that $n_1^{inf} < \cdots < n_P^{inf}$.

If $P = 1$, that is, if graph $\mathcal{G}$ is Eulerian, the procedure returns the schedule defined by circuit C_1 (Lines 17 and 25). Note that the schedule corresponds to the sequence in which requests are served which, in turn, is the sequence in which the corresponding arcs are traversed in the Eulerian graph. Otherwise, in Lines 17 to 24, the algorithm connects the P circuits using arcs $(n_p^{inf}, n_{p+1}^{inf})$ to obtain a single circuit Λ. Then, it returns the schedule obtained from this circuit (Line 25). Theorem 1 proves that in both cases the schedules are optimal.

The intuition behind the algorithm is based on the observation that the optimal number of tours is either N^{sup} or $N^{sup}+1$ and that adding dummy unit-length requests does not change the optimal solution value, as long as the value of N^{sup} does not change. This allows to model the problem as determining an Eulerian circuit in a graph augmented with the dummy requests.

The various lemmas that are needed to prove the main result in Theorem 1 are now proved. In what follows, multisets $\mathcal{R}^{(1)}$ and $\mathcal{R}^{(2)}$, graph $\mathcal{G}$ and circuit Λ are those obtained from Algorithm 1. Two illustrative examples follow.

Lemma 1. *$T_{OPT} \geq N^{sup}$.*

Proof. Since a vehicle of capacity one is considered, conflicting requests must be served in different tours. The N^{sup} requests that cover the same segment must then be active on N^{sup} different tours. It follows that the number of tour in any feasible solution is at least N^{sup}. □

Lemma 2. *Multiset $\mathcal{R}^{(1)}$ is such that $z(\mathcal{R}^{(1)}) = z(\mathcal{R})$.*

Proof. We call $\Lambda^*(\mathcal{R})$ the optimal schedule when the request set is $\mathcal{R}$. From Lemma 1, we know that the vehicle performs at least N^{sup} tours in $\Lambda^*(\mathcal{R})$. Furthermore,

we know that every segment $[j, j+1]$ is covered exactly $N(j, j+1)$ times in request multiset $\mathcal{R}$. It means that the vehicle is not active on at least $N^{sup} - N(j, j+1)$ tours in schedule $\Lambda^*(\mathcal{R})$ when it traverses segment $[j, j+1]$. A feasible schedule for request multiset $\mathcal{R}^{(1)}$ can be constructed from $\Lambda^*(\mathcal{R})$ by serving the new requests $(j, j+1)$ when the vehicle is not active. This schedule has the same cost than $\Lambda^*(\mathcal{R})$, that is, $z(\mathcal{R})$. It follows that $z(\mathcal{R}^{(1)}) \leq z(\mathcal{R})$. Trivially, as $\mathcal{R} \subseteq \mathcal{R}^{(1)}$, we have that $z(\mathcal{R}^{(1)}) \geq z(\mathcal{R})$. Thus, $z(\mathcal{R}^{(1)}) = z(\mathcal{R})$. □

Lemma 3. *If $N^*(0) = N^{sup}$ then $T_{OPT} \geq N^{sup} + 1$.*

Proof. If $N^*(0) = N^{sup}$, N^{sup} requests cover the depot and have to be active at the depot on different tours. Given a feasible solution, the first active request at the depot cannot be finished before tour number 2, the second before tour number 3, and, by a simple induction, active request number N^{sup} before tour number $N^{sup}+1$. It implies $T_{OPT} \geq N^{sup} + 1$. □

Lemma 4. *Multisets $\mathcal{R}^{(1)}$ and $\mathcal{R}^{(2)}$ are such that $z(\mathcal{R}^{(2)}) = z(\mathcal{R}^{(1)})$.*

Proof. If $N^*(0) < N^{sup}$, $\mathcal{R}^{(2)} = \mathcal{R}^{(1)}$. Otherwise, we can follow exactly the same proof as in Lemma 2. With request multiset $\mathcal{R}^{(2)}$, we know, thanks to Lemma 3, that the vehicle performs at least $N^{sup}+1$ tours. So, compared to $\mathcal{R}^{(1)}$, an additional request can be added for each segment $[j, j+1]$ without increasing the solution cost. □

Lemma 5. *Graph $\mathcal{G}$ is semi-Eulerian.*

Proof. For each node j of graph $\mathcal{G}$, let $d^+_G(j)$ be the number of outgoing arcs and $d^-_G(j)$ be the number of ingoing arcs. To show that the graph is semi-Eulerian, it suffices to prove $d^+_G(j) = d^-_G(j)$ for $j = 0, \ldots, m-1$. We know that every segment $[j, j+1]$ is covered by the same number N of requests from $\mathcal{R}^{(2)}$ ($N = N^{sup}$ or $N^{sup}+1$, depending on whether $N^*(0) < N^{sup}$ or $N^*(0) = N^{sup}$, respectively). On the segment that follows j, these N requests are made up of $d^+_G(j)$ requests whose source node is j and $N^*(j)$ requests that cover j (and therefore cover the segment as well). On the segment that precedes j, they include the set of $d^-_G(j)$ requests whose target node is j and the $N^*(j)$ requests that cover j. Therefore, for any $j \in \{0, \ldots, m-1\}$, $d^+_G(j) + N^*(j) = N = d^-_G(j) + N^*(j)$, which proves the lemma. □

Lemma 6. *If $N^*(0) = N^{sup}$, graph $\mathcal{G}$ is Eulerian.*

Proof. If $N^*(0) = N^{sup}$, $\mathcal{R}^{(2)}$ contains at least once each request $(j, j+1)$ for $j = 0, \ldots, m-1$. These requests create a directed circuit in $\mathcal{G}$ containing all the nodes. Graph $\mathcal{G}$ is then strongly connected. As Lemma 5 states that $\mathcal{G}$ is semi-Eulerian, then $\mathcal{G}$ is Eulerian. □

Theorem 1. *Schedule Λ is optimal. Three cases can be distinguished:*

1. *If* $N^*(0) = N^{sup}$, $T_{OPT} = N^{sup} + 1$
2. *If* $N^*(0) < N^{sup}$ *and graph* $\mathcal{G}$ *is Eulerian (algorithm decomposeEulerian*$(\mathcal{G})$ *returns a single circuit),* $T_{OPT} = N^{sup}$
3. *If* $N^*(0) < N^{sup}$ *and graph* $\mathcal{G}$ *is not Eulerian (algorithm decomposeEulerian*$(\mathcal{G})$ *returns several circuits),* $T_{OPT} = N^{sup} + 1$

Proof. The three cases are proven separately:

1. If $N^*(0) = N^{sup}$, $\mathcal{G}$ is Eulerian. Every segment $[j, j+1]$ is covered exactly $N^{sup}+1$ times in $\mathcal{R}^{(2)}$, so the sum of request lengths is $(N^{sup}+1) \times L$. Circuit Λ contains an arc for each request in $\mathcal{R}^{(2)}$. Its total length is also $(N^{sup}+1) \times L$. Thus, $T_{OPT} \leq N^{sup} + 1$. From Lemma 1, we have $T_{OPT} \geq N^{sup} + 1$. This demonstrates that $T_{OPT} = N^{sup} + 1$ and that schedule Λ is optimal.
2. If $N^*(0) < N^{sup}$ and graph $\mathcal{G}$ is Eulerian, we can apply exactly the same proof, except that every segment is now covered N^{sup} times. We obtain $T_{OPT} = N^{sup}$ and schedule Λ is optimal.
3. If $N^*(0) < N^{sup}$ and graph $\mathcal{G}$ is not Eulerian, $T_{OPT} \geq N^{sup} + 1$. The proof is by contradiction. Assume $T_{OPT} \leq N^{sup}$. From Lemma 1 it means $T_{OPT} = N^{sup}$, that is, the optimal schedule exactly covers the arcs in $\mathcal{G}$. However, this contradicts the fact that $\mathcal{G}$ is not connected. Thus, $T_{OPT} \geq N^{sup} + 1$. Schedule Λ can now shown to be optimal. The total length of this schedule is $N^{sup} \times L + \sum_{1 \leq p \leq P-1} \delta_{n_p^{inf}, n_{p+1}^{inf}} + \delta_{n_P^{inf}, n_1^{inf}} = N^{sup} \times L + L = (N^{sup} + 1) \times L$. Thus, the number of tours is $N^{sup} + 1$ and it is optimal.

□

Using Hierholzer's algorithm , procedure *decomposeEulerian*$(\mathcal{G})$ can be implemented with a complexity $O(N^{sup} \times m)$ (see, for example, Jungnickel, 2013). The different loops of the algorithm (to compute values $N(j, j+1)$ and $N^*(j)$, to construct $\mathcal{R}^{(1)}$ and $\mathcal{R}^{(2)}$, to obtain Λ) all have either the same complexity or a lower complexity. The overall complexity of the algorithm is thus $O(N^{sup} \times m)$. Seeing that $N^{sup} \leq n$, it proves that problem $1, 1|sd, u|CLT$ is polyniomally solvable.

The execution of the algorithm is illustrated with the following two examples.

Example 1

Let us consider a ring with $m = 5$ equidistant stations (numbered from 0 to 4) and four requests: $\mathcal{R} = \{(4, 2), (2, 3), (1, 3), (3, 1)\}$ (see Figure 2.3). On this example:

- $N(0,1) = N(1,2) = N(2,3) = N(4,0) = 2$, $N(3,4) = 1$, $N^{sup} = 2$ and $N^*(0) = 2$
- $\mathcal{R}^{(1)} = \mathcal{R} \cup \{(3,4)\}$ (see Figure 2.4)
- $\mathcal{R}^{(2)} = \mathcal{R}^{(1)} \cup \{(0,1),(1,2),(2,3),(3,4),(4,0)\}$ (see Figure 2.4)

Based on multiset $\mathcal{R}^{(2)}$, we obtain graph $\mathcal{G}$ represented on Figure 2.5. Applying Hierholzer's algorithm provides a single optimal circuit (the graph is Eulerian), *e.g.*: $0 \to 1 \to 2 \to 3 \to 1 \to 3 \to 4 \to 2 \to 3 \to 4 \to 0$. This circuit gives the schedule $(2,3),(3,1),(1,3),(4,2)$ which is completed in three tours on the ring.

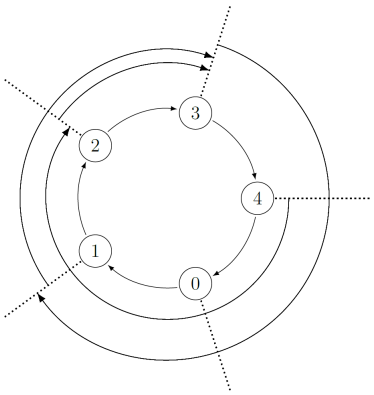
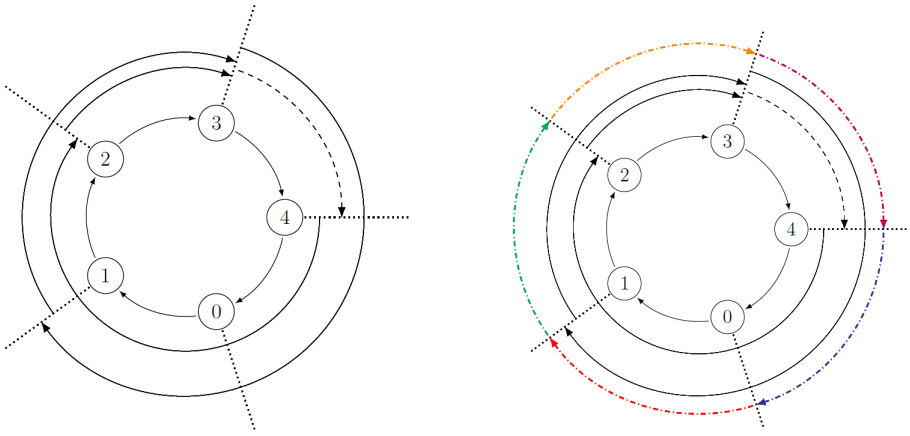


Figure 2.3: Example 1: Ring and initial requests ($\mathcal{R}$)



(a) Request multiset $\mathcal{R}^{(1)}$ (b) Request multiset $\mathcal{R}^{(2)}$

Figure 2.4: Addition of new requests. The dashed arc denotes the request added in lines 6-8, while the dash-dotted arcs denote the requests added in lines 9-14 of Algorithm 1.

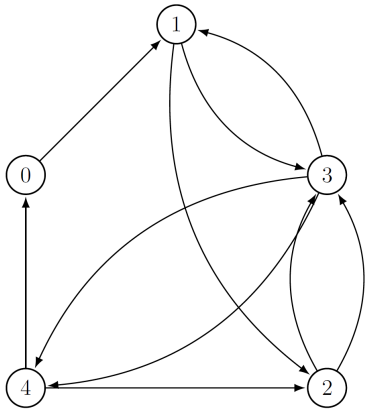


Figure 2.5: Graph $\mathcal{G}$ of Example 1

Example 2

Consider the same ring and a new request set: $\mathcal{R} = \{(0,4),(4,0)(1,3),$ $(3,1)\}$ (see Figure 2.6):

- $N(0,1) = N(1,2) = N(2,3) = N(3,4) = N(4,0) = 2$, $N^{sup} = 2$ and $N^*(0) = 1$
- no new requests are added: $\mathcal{R}^{(2)} = \mathcal{R}^{(1)} = \mathcal{R}$

Graph $\mathcal{G}$ is depicted on Figure 2.7. Applying Hierholzer's algorithm provides two Eulerian circuits $C_1 = 0 \to 4 \to 0$ and $C_2 = 1 \to 3 \to 1$. These two circuits are reconnected with arcs $(0,1)$ and $(1,0)$ to form $\Lambda = 0 \to 4 \to 0 \to 1 \to 3 \to 1 \to 0$. The vehicle performs 3 tours and the schedule is $(0,4),(4,0),(1,3),(3,1)$.

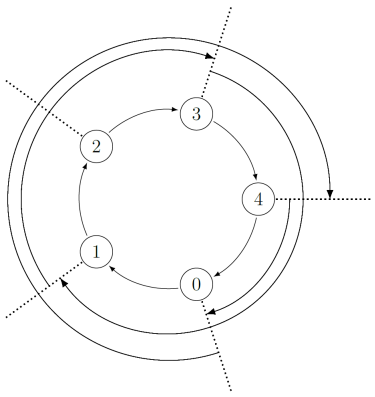


Figure 2.6: Example 2: Ring and initial requests ($\mathcal{R}$)

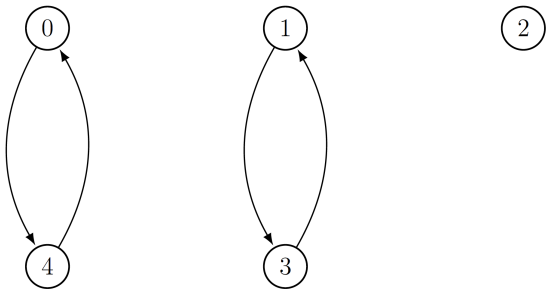


Figure 2.7: Graph $\mathcal{G}$ of Example 2

2.3.2 Problems $\alpha|sd, u|CLT$, with $\alpha = V, 1$ or $1, Q$ or V, Q or V, Q_v

In this section different generalizations of problem $1, 1|sd, u|CLT$ are proven NP-hard.

The problem with multiple vehicles of capacity 1 is is first shown NP-hard (problem $V, 1|sd, u|CLT$). Because of the length of the proof, the proof is omitted here and can be found in Chapter 3. Then, this problem is proven equivalent to the problem faced when the fleet is composed of a single vehicle of non-unitary capacity (problem $1, Q|sd, u|CLT$), which demonstrates that $1, Q|sd, u|CLT$ is also NP-hard. These results show that $V, Q|sd, u|CLT$ and $V, Q_v|sd, u|CLT$, that generalize the two others, are NP-hard.

Theorem 2. *Problem $V, 1|sd, u|CLT$ is NP-hard.*

See proof in Chapter 3.

What follows is a very simple example of an instance of $V, 1|sd, u|CLT$ where the optimal value CLT is greater than $\lceil \frac{CLT^*}{V} \rceil$, where CLT^* is the optimal value of an instance of $1, 1|sd, u|CLT$ with the same set of requests. Consider an instance of $1, 1|sd, u|CLT$ where the set of requests $\mathcal{R}$ is composed of the only request $(m-1, 1)$ (with $m \geq 3$). The optimal value CLT^* is 2. Consider an instance of $V, 1|sd, u|CLT$ with the same request and $V = 2$. The optimal value CLT does not change, and thus it holds $CLT = 2 > 1 = \lceil \frac{CLT^*}{V} \rceil$.

Theorem 3. *Problem $1, Q|sd, u|CLT$ is NP-hard.*

Proof. Let us consider an instance of $1, Q|sd, u|CLT$ and the equivalent instance of $V, 1|sd, u|CLT$ where the vehicle of capacity Q is replaced by V vehicles of capacity 1, with $V = Q$. To prove that $1, Q|sd, u|CLT$ is NP-hard, it must be proven that any feasible solution $\Lambda^{1,Q}$ of the former problem can be transformed to a same-cost feasible solution $\Lambda^{V,1}$ of the latter, and vice-versa.

Considering a feasible schedule $\Lambda^{1,Q}$, we assign a number v_i between 1 and Q to all requests $i \in \mathcal{R}$, so that two requests active at the same time have different numbers. This numbering always exists as, when a request starts, at most $Q-1$ other requests are active and so, a number between 1 and Q is available. Then, we build schedule $\Lambda^{V,1}$ by starting all requests as in $\Lambda^{1,Q}$ and by assigning every request to vehicle number v_i. The schedule is feasible because two requests with the same number are not active at the same time and because $v_i \leq V$ for all requests. Both schedules $\Lambda^{1,Q}$ and $\Lambda^{V,1}$ are feasible for their respective problem and have the same closing time. Starting with a feasible schedule $\Lambda^{V,1}$, the reverse transformation can be applied exactly the same, with the same conclusion. The two problems having the same set of feasible solutions with equivalent costs, they have the same complexity. It permits to conclude that problem $1, Q|sd, u|CLT$ is NP-hard. □

2.3.3 An ILP formulation for problem $V, Q|sd, u|CLT$

As problem $V, Q|sd, u|CLT$ is NP-hard, a mathematical formulation is proposed that could allow solving it. The formulation makes use of the following notation, in addition to the one introduced in previous sections.

- K: maximum tour number at which a request should be started
- $\mathcal{R}[j, j+1]$: set of requests in $\mathcal{R}$ that cover segment $[j, j+1]$.
- c_{ik}: cost of inserting request $i \in \mathcal{R}$ if it is started in tour number k

$$c_{ik} = \begin{cases} k & \text{if } s_i < t_i \\ k+1 & \text{otherwise} \end{cases}$$

To calculate an upper bound for K, finding a feasible solution is enough. Requests can be ordered in the increasing order of their pickup station, and started in consecutive tours. Then, the last request would start at tour n. Therefore, $K = n$.

We introduce the following decision variables:

$$\begin{aligned} x_{ik} &= \begin{cases} 1 & \text{if request } i \text{ is started in tour k} \\ 0 & \text{otherwise} \end{cases} \\ CLT &= \text{closing time.} \end{aligned}$$

with $(1 \leq i \leq n, 1 \leq k \leq K)$.

Note that the closing time corresponds to the maximum number of tours traversed by each vehicle multiplied by the length of the ring L. As L is a constant, then minimizing CLT corresponds to minimizing the maximum number of tours traversed by each vehicle. Thus, in the following we refer to CLT as the latter number.

The integer linear program is then:

$$\min CLT \tag{2.2}$$

s.t.:

$$\sum_{\{i \in \mathcal{R}[j,j+1], s_i \leq j\}} x_{ik} + \sum_{\{i \in \mathcal{R}[j,j+1], s_i > j+1\}} x_{ik-1} \leq VQ \tag{2.3}$$

$$\sum_{k \in \mathcal{K}} x_{ik} = 1 \tag{2.4}$$

$$CLT \geq \sum_{k \in \mathcal{K}} c_{ik} x_{ik} \tag{2.5}$$

$$x_{ik} \in \{0, 1\} \tag{2.6}$$

$$CLT \geq 0 \tag{2.7}$$

with $(1 \leq i \leq n, 0 \leq j \leq m-1, 1 \leq k \leq K)$.

The objective function minimizes the closing time, expressed in number of tours. Constraints (2.3) make sure that the number of active requests never exceeds the total capacity of the V vehicles. For each segment $[j, j+1]$ and each tour k, this number is evaluated by counting the active requests that started in the tour before station j and those that started in the preceding tour after station $j+1$. Constraints (2.4) make sure that every request is served. Constraints (2.5) compute the maximal tour number CLT. Constraints (2.6)-(2.7) define decision variables.

2.4 Problems with unitary requests and release/-due dates

This section focuses on the class of problems where each request $i \in R$ is unitary and is subject to a release date r_i and a due date d_i. Note that in the context of the single-direction (sd) constraint, release dates and due dates can be expressed as the smallest and the highest tour number in which the request can be served.

2.4.1 Problem $1,1|sd,u,r_i,d_i|CLT$

This section is devoted to the problem where a single vehicle of capacity one has to serve the requests. The problem is the same as in Section 2.3.1 with the addition of release dates and due dates. We will see that, when having release dates and due dates, the problem becomes NP-hard. We proceed by reduction from a variant of the list coloring problem, called (γ, μ)-coloring problem, introduced in Bonomo et al., 2009 and shown to be NP-hard.

Definition 8. *Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and functions $\gamma, \mu : \mathcal{V} \to \mathbb{N}$ such that $\gamma(v) \leq \mu(v)$ for every $v \in \mathcal{V}$, we say that $\mathcal{G}$ is (γ, μ)-colorable if there exists a function $f : \mathcal{V} \to \mathbb{N}$ of $\mathcal{G}$ such that $\gamma(v) \leq f(v) \leq \mu(v)$ for every $v \in \mathcal{V}$, and $f(v) \neq f(v')$ for every $(v, v') \in \mathcal{E}$.*

In Bonomo et al., 2009 it is shown that this problem generalizes the graph coloring problem and is NP-hard, also in the case where the underlying graph is an interval graph. An interval graph is the intersection graph of a set of intervals over the real line. As shown later, this result is used when proving that problem $1, 1|sd, u, r_i, d_i|CLT$ is NP-hard.

Theorem 4. *Problem $1, 1|sd, u, r_i, d_i|CLT$ is NP-hard.*

Proof. The main argument in the proof is that the constraints implied by release dates and due dates can be equivalently expressed as constraints on the lowest and largest tour number on which a request can be started. The starting tour of a request can then be interpreted as a color, bounded by these two limits. Given that fact, we will see how to transform an instance of the decision version of the (γ, μ)-coloring problem on an interval graph into an instance of the decision version of $1, 1|sd, u, r_i, d_i|CLT$.

Consider an instance I_c of the decision version of the (γ, μ)-coloring problem on an interval graph. $\mathcal{A} = \{(a_1, b_1), ..., (a_n, b_n)\}$ is a set of n intervals on a real line with $a < b$ for each $(a, b) \in \mathcal{A}$. An interval graph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ is constructed from $\mathcal{A}$ by introducing a vertex v_i for each interval $(a_i, b_i) \in \mathcal{A}$ and by adding an edge (v_i, v_j) each time intervals (a_i, b_i) and (a_j, b_j) overlap. Let γ and μ be two functions that map vertices in $\mathcal{V}$ into natural numbers, such that $\gamma(v) \leq \mu(v)$ for each $v \in \mathcal{V}$. Instance I_c consists in deciding if there exists a coloring function $f : \mathcal{V} \to \mathbb{N}$ such that $\gamma(v) \leq f(v) \leq \mu(v)$ for each $v \in \mathcal{V}$ and $f(v) \neq f(v')$ for $(v, v') \in \mathcal{E}$.

We build an instance I_s of the decision version of $1, 1|sd, u, r_i, d_i|CLT$ as follows. Let $\mathcal{D}$ denote the set of interval extremities: $\mathcal{D} = \{a_i, b_i : i = 1, ..., n\}$. We sort $\mathcal{D}$ in increasing order and introduce a station in the ring for every element in $\mathcal{D}$, in this order. We complete the ring with station 0. This way, the number m of stations is at most two times the number n of intervals plus the depot ($m \leq 2n + 1$). We define unitary distances between successive stations, which gives a tour length $L = |\mathcal{D}| + 1$. We introduce a request in $\mathcal{R}$ for every interval in $\mathcal{A}$. Given interval (a_i, b_i), request i is defined as follows: s_i is the station obtained from extremity a_i, t_i is the station obtained from extremity b_i, r_i is set to $L \times \gamma(v_i)$ and $d_i = L \times (\mu(v_i) + 1))$. Basically, this means that request i has to be active between the beginning of tour number $\gamma(v_i)$ and the end of tour number $\mu(v_i)$. Note that no request covers the depot, so two requests will be scheduled in the same tour if and only if they do not overlap. Note also that this construction is polynomial.

We claim that $\mathcal{H}$ is (γ, μ)-colorable if and only if I_s admits a feasible schedule. A feasible schedule for I_s is a function $f : \mathcal{R} \to \mathbb{N}$ that assigns each request $i \in \mathcal{R}$ to a

tour (that is the tour in which the request is served) so that requests do not overlap and that the tour number of request i lies between the lowest possible tour number and highest possible tour number given by the release and due dates, that is, $\gamma(v_i)$ and $\mu(v_i)$. More formally, function f has to be such that:

- $f(i_1) = f(i_2) \implies i_1$ and i_2 do not overlap $\forall i_1, i_2 \in \mathcal{R}$
- $\gamma(v_i) \leq f(i) \leq \mu(v_i)\ \forall i \in \mathcal{R}$

Assume first that $\mathcal{H}$ is (γ, μ)-colorable. This means that a function $g : V \to \mathbb{N}$ is a coloring for $\mathcal{H}$. Let $f : R \to \mathbb{N}$ be a function such that $f(i) = g(v_i)$, $v_i \in V$. It is easy to see that f is a feasible schedule for I_s.

On the other hand, assume that I_s admits a feasible schedule f. Let $g : \mathcal{V} \to \mathbb{R}$ be a function such that $g(v_i) = f(i)$, $(a_i, b_i) \in \mathcal{A}$. It is easy to see that g is a coloring function for $\mathcal{H}$. In fact, $g(v_{i_1}) = g(v_{i_2})$ means that requests i_1 and i_2 do not overlap, *i.e.,* edge $(v_{i_1}, v_{i_2}) \notin \mathcal{E}$. In addition, from the definition of r_i and d_i, $\gamma(v_i) \leq g(v_i) \leq \mu(v_i)$.

It shows that solving the (γ, μ)-coloring problem on an interval graph amounts to finding a feasible solution to an instance of problem
$1, 1|sd, u, r_i, d_i|CLT$. This proves that problem $1, 1|sd, u, r_i, d_i|CLT$ is NP-hard. □

Given the result of Theorem 4, it follows that all problems with release and due dates, monodirectional and where the objective function is the minimization of CLT are NP-Hard.

The following section contains a mathematical formulation for problem $V, Q|sd, u, r_i, d_i|CLT$ which can be used to solve also problems
$\alpha|sd, u, r_i, d_i|CLT$, with $\alpha = 1, 1$ or $1, Q$ or $V, 1$, and that could be easily adapted to solve problems with heterogeneous fleet of vehicles.

2.4.2 An ILP formulation for problem $V, Q|sd, u, r_i, d_i|CLT$

In addition to notation defined in previous sections, we define $\mathcal{K}_i$ as the set of tour numbers in which request i can be started

$$\mathcal{K}_i = \{k \in \mathbb{N} : (k-1)L + \delta_{0,s_i} \geq r_i \text{ and } (k-1)L + \delta_{0,s_i} + \delta_{s_i,t_i} \leq d_i\}$$

Also, we tighten K, that is, the tour number upper bound for the last started request, by taking account of due dates:

$$K = \max_{1 \leq i \leq n} \left(\left\lceil \frac{d_i}{L} \right\rceil \right)$$

Decision variables are the same as in model (2.2)-(2.7). The formulation is given by (2.2)-(2.3), (2.5)-(2.7) and the following modification of constraints (2.4):

$$\sum_{k \in \mathcal{K}_i} x_{ik} = 1 \qquad (1 \leq i \leq n) \tag{2.8}$$

Constraints (2.8) make sure that every request is served and that release and due dates are respected.

2.5 Problems with non-unitary requests

This section considers the extension of previous problems to non-unitary requests. Note that in this context, the case $Q = 1$ does not make sense. Remembering that $1, Q|sd, u|CLT$ is NP-hard (see Section 2.3.2), we can conclude that all problems investigated in this section are NP-hard.

2.5.1 Alternative complexity proof for problem $1, Q|sd|CLT$

Though theorems 2 and 3 show that problem $1, Q|sd|CLT$ is NP-hard, the proof appears overly complex when requests are not necessarily unitary. In this section, a simpler proof based on a reduction from the Bin Packing Problem (BPP) is proposed.

The BPP is defined as follows. We are given a finite set $\mathcal{O}$ of n objects and K bins. Let $W \in \mathbb{Z}^+$ be the bin capacity and w_o be the weight of each object $o \in \mathcal{O}$, with $w_o \leq W$. The problem consists in determining whether a partition of $\mathcal{O}$ in K disjoint subsets $\mathcal{O}_1, \ldots, \mathcal{O}_K$ exists such that the sum of item weights in each subset $\mathcal{O}_k$ is at most W. This problem is NP-complete.

Theorem 5. *Problem $1, Q|sd|CLT$ is NP-hard*

Proof. We consider the decision version of problem $1, Q|sd|CLT$, *i.e.*, the problem of determining, given an integer C, whether a feasible schedule exists with $CLT \leq C$.

Let us consider an instance I_B for the BPP, with a set $\mathcal{O}$ of objects and K bins. We construct an instance I_S for $1, Q|sd|CLT$ as follows. We define a ring with two stations $\mathcal{S} = \{0, 1\}$. For each object $o \in \mathcal{O}$, we introduce a request i in $\mathcal{R}$, with $s_i = 0$, $t_i = 1$ and $q_i = w_o$. The capacity Q of the vehicle is set to bin capacity W. We finally set $C = K$. In instance I_S, all requests overlap, hence, a schedule Λ is feasible if and only if:

- in each tour, the sum of the demands of the requests started in the tour is not greater than Q,
- the schedule contains at most C tours.

We show that I_B is feasible if and only if I_S is. Assume first I_B feasible. We build a schedule Λ for I_S by executing the requests of the K bins in K separate tours. Conversely, let us assume I_S feasible. We consider a feasible schedule Λ. We build a solution for I_B by assigning all requests starting in the same tour to the same bin. This shows that solving the BPP can be reduced to solving the decision version of $1, Q|sd|CLT$, with a polynomial reduction, which proves that $1, Q|sd|CLT$ is NP-hard. □

2.5.2 An ILP model

In this section, a mathematical formulation for problem $V, Q|sd, r_i, d_i|CLT$ is presented. This formulation can also be used to solve problems $1, 1|sd, r_i, d_i|CLT$, $1, Q|sd, r_i, d_i|CLT$ and $V, 1|sd, r_i, d_i|CLT$.

This formulation extends the formulations presented in Section 2.3.3 and 2.4.2. The notation introduced in these sections is used. Tour assignment variables takes a third index to take into account the vehicle that serves a request:

$$x_{ikv} = \begin{cases} 1 & \text{if request } i \text{ is started in tour } k \text{ by vehicle } v \\ 0 & \text{otherwise} \end{cases}$$

Recall that q_i is the demand associated with request i. The integer linear program is then:

$$\min CLT \tag{2.9}$$

s.t.:

$$\sum_{\{i \in \mathcal{R}[j,j+1], s_i \leq j\}} q_i x_{ikv} + \sum_{\{i \in \mathcal{R}[j,j+1], s_i > j+1\}} q_i x_{ik-1v} \leq Q \tag{2.10}$$

$$\sum_{v \in \{1,\dots,V\}} \sum_{k \in \mathcal{K}_i} x_{ikv} = 1 \tag{2.11}$$

$$CLT \geq \sum_{v \in \{1,\dots,V\}} \sum_{k \in \mathcal{K}_i} c_{ik} x_{ikv} \tag{2.12}$$

$$x_{ikv} \in \{0, 1\} \tag{2.13}$$

$$CLT \geq 0 \tag{2.14}$$

with $1 \leq i \leq n, 0 \leq j \leq m-1, 1 \leq k \leq K, 1 \leq v \leq V$.

The objective function minimizes the closing time, expressed as number of tours, over all vehicles. Constraints (2.10) ensure that vehicle capacities are satisfied. They disaggregate constraints (2.3) for each vehicle. Constraints (2.11) make sure that every requests is served, at an acceptable time. Constraints (2.12) compute CLT. Constraints (2.13)-(2.14) define variables domain.

The following are symmetry breaking constraints to strengthen formulation (2.9)–(2.14). The first set of constraints ranks vehicles according to the number of satisfied requests: the smallest the vehicle index, the largest the number of satisfied requests

$$\sum_{i=1}^{n} \sum_{k \in \mathcal{K}_i} x_{ikv} \geq \sum_{i=1}^{n} \sum_{k \in \mathcal{K}_i} x_{ikv+1} \qquad (1 \leq v \leq V-1). \tag{2.15}$$

The second set imposes that the first requests are served by the first vehicles: the i first requests have to be assigned to vehicles in set $\{1, \ldots, i\}$:

$$\sum_{u \in \{1,\ldots,j\}} \sum_{v \in \{1,\ldots,j\}} \sum_{k \in \mathcal{K}_i} x_{ukv} = j \qquad (1 \leq j \leq \min(V, n)). \tag{2.16}$$

Note that (2.15) and (2.16) cannot be used simultaneously.

The impact of these constraints was evaluated computationally and a slight improvement was noticed, in terms of computing times, over the formulation without symmetry breaking constraints. In particular, the most effective constraints are the ones given by inequalities (2.16). Thus, the computational results presented in the following include these constraints.

2.6 Computational tests

It is now presented the set of experiments made in order to evaluate the efficacy of the formulations presented above. The formulations are solved through CPLEX 12.9.0 on a computer equipped with an Intel Core i7-9700 processor and 32GB of RAM. For all instances a time limit of 30 minutes was set. As shown in the results, all instances were solved (either to optimality or by proving infeasibility) within this time limit.

2.6.1 Instance sets

As far as is known, the problems investigated in this chapter are new and no benchmark instances exist. Random instances are generated as follows. All random numbers are chosen from uniform distributions.

A ring with $m = 10$ stations is considered. The number of requests is chosen in the set $\{20, 40, 80, 160\}$, and 5 instances are generated for each number of requests. The pickup station of a request is randomly taken in $\{0, \ldots, m-1\}$ and the target station in $\{0, \ldots, m-1\} \setminus \{s_i\}$. For each instance, distances between consecutive stations are randomly drawn in $[1, 10]$. This gives a total of 20 combinations of the parameters mentioned above.

Every instance is replicated several times with a different fleet of vehicles and demand values. The following values for V and Q are considered: $V \in \{1, 2, 3\}$, $Q \in \{1, 2, 4, 8\}$. When demands are not unitary, they are generated in interval $[1, Q]$. For each of the 20 combinations mentioned above, this gives 12 fleet/demand compositions for unitary and non-unitary demands, respectively.

When needed, release dates and due dates are defined based on the following observations. The expected length of a request is approximately $\frac{L}{2}$. The expected demand is approximately $\frac{Q}{2}$ when demands are non-unitary or exactly 1 when they are. So, every request approximately generates an expected workload $\frac{L}{2} \times \frac{Q}{2}$ in unit-of-distance $\times$ unit-of-demand for non-unitary demands and an expected workload $\frac{L}{2}$ otherwise. The total expected workload is thus $\frac{L}{2} \times \frac{Q}{2} \times n$ or $\frac{L}{2} \times n$, respectively. The fleet brings a total capacity $CLT \times V \times Q$ in the same unit. With a capacity fully exploited, CLT should thus approach $\frac{nL}{4V}$ when demands are not unitary, $\frac{nL}{2QV}$ otherwise. Let C denote this value.

Then:

- Release dates are randomly generated in intervals $[0, 1.5C]$ (tight) or $[0, 2C]$ (wide)
- Due dates are generated in intervals $[r_i + L, r_i + L + C]$ (tight) or $[r_i + L, r_i + L + 1.5C]$ (wide)

For the 20×24 combinations above, this gives four alternatives for release and due dates: t-t, t-w, w-t, w-w, where t and w stand for tight and wide, respectively.

For each instance constructed this way, two variants are finally considered: the original instance and the instance where due dates are relaxed. The reason behind the latter is to investigate the impact of due dates on the problem tractability. Also, it might correspond to a realistic situation where, for example, goods have to be moved between stations to prepare the planning for the next day. In this case, goods are ready at a given time at the pickup station (the release date) but there is no restriction on the time at which they should be available at the delivery station. Instances are available at https://github.com/manueltrotta/PDP-R-instances-and-results.

To calculate value K in the case with release dates and no due date, it suffices to notice that at the beginning of tour $\left\lceil \frac{2C}{L} \right\rceil + 1$ all requests are ready to be processed

(release dates are passed). Then, the reasoning of Section 2.3.3 can be applied to set $K = \left\lceil \frac{2C}{L} \right\rceil + n$.

2.6.2 Greedy algorithm

In order to evaluate the benefits of solving the problems to optimality, the following simple greedy algorithm is designed that is easy to implement and that would be easy to understand for practitioners. The solutions provided by the greedy algorithm are then compared with the optimal solutions. The greedy algorithm works as follows.

When a vehicle reaches a station that is the pickup station of at least one request and this request can be started, it is started. If several requests can be started at the same time, ties are broken by giving priority to requests according to the following criteria, taken hierarchically:

- earliest due date first (due date is considered infinite if no due date is defined);
- largest demand first;
- longest distance to reach the delivery node first.

Note that, in case of instances with unit demands and no release/due dates, only the third criterion is used.

2.6.3 Computational results for $V, Q|sd, u|CLT$

In Table 2.1 results for problem $V, Q|sd, u|CLT$ are reported. All results are averaged over the 5 instances with the same value of n, V and Q, considering feasible instances only. Detailed results are available at https://github.com/manueltrotta/PDP-R-instances-and-results. Column *CPU(s)* reports the solution time of the formulation in Section 2.3.3, in seconds. Column *feas* reports the number of feasible instances. Column *greedy* reports the number of instances for which the greedy algorithm found a feasible solution. Column T_{OPT} reports the value of the optimal solution (closing time expressed in number of tours). Column *gapGr* gives the average percentage gap of the solutions found with the greedy algorithm with respect to the optimal solution.

n	V	Q	CPU(s)	feas	greedy	CLT	gapGr(%)	n	V	Q	CPU(s)	feas	greedy	CLT	gapGr(%)
20	1	1	0.1	5	5	13.0	4.5	80	1	1	2.5	5	5	46.4	1.3
20	1	2	0.0	5	5	6.8	5.4	80	1	2	1.2	5	5	23.4	1.6
20	1	4	0.0	5	5	3.8	11.7	80	1	4	0.6	5	5	12.0	1.7
20	1	8	0.0	5	5	2.6	20.0	80	1	8	0.2	5	5	6.4	12.9
20	2	1	0.1	5	5	6.8	8.7	80	2	1	2.5	5	5	23.4	1.6
20	2	2	0.0	5	5	3.8	11.7	80	2	2	1.3	5	5	12.0	5.0
20	2	4	0.0	5	5	2.6	20.0	80	2	4	0.6	5	5	6.4	6.2
20	2	8	0.0	5	5	2.0	0.0	80	2	8	0.4	5	5	3.4	25.0
20	3	1	0.1	5	5	4.8	9.0	80	3	1	3.8	5	5	15.8	3.8
20	3	2	0.0	5	5	3.0	6.7	80	3	2	1.7	5	5	8.2	9.7
20	3	4	0.0	5	5	2.2	10.0	80	3	4	0.8	5	5	4.4	19.0
20	3	8	0.0	5	5	2.0	0.0	80	3	8	0.3	5	5	3.0	6.7
40	1	1	0.3	5	5	25.4	0.7	160	1	1	17.6	5	5	87.4	0.2
40	1	2	0.2	5	5	13.2	6.1	160	1	2	8.2	5	5	44.2	0.9
40	1	4	0.1	5	5	7.0	2.9	160	1	4	5.3	5	5	22.2	2.8
40	1	8	0.1	5	5	4.0	10.0	160	1	8	2.0	5	5	11.4	5.5
40	2	1	0.4	5	5	13.2	4.7	160	2	1	30.9	5	5	44.2	0.5
40	2	2	0.2	5	5	7.0	9.0	160	2	2	19.9	5	5	22.2	1.9
40	2	4	0.1	5	5	4.0	5.0	160	2	4	6.6	5	5	11.4	5.5
40	2	8	0.1	5	5	2.6	20.0	160	2	8	3.3	5	5	6.0	16.7
40	3	1	0.4	5	5	9.0	4.2	160	3	1	39.9	5	5	29.4	2.1
40	3	2	0.2	5	5	5.0	8.0	160	3	2	18.8	5	5	15.0	5.4
40	3	4	0.1	5	5	3.0	13.3	160	3	4	9.9	5	5	7.8	10.4
40	3	8	0.1	5	5	2.0	10.0	160	3	8	4.3	5	5	4.0	25.0

Table 2.1: Problem $V, Q|sd, u|CLT$

We can observe that the formulation in Section 2.3.3 is extremely effective. In fact, the computational time is always smaller than 40 seconds. The computational time increases with the number of requests, as expected, and decreases with the vehicle capacity. Also, the solution value decreases as the capacity increases, as expected. The performance of the greedy algorithm highly depends on vehicle capacity: the larger the capacity, the worse is the performance. This might be due to the fact that, when increasing the capacity, the chances of making a bad choice of assignment of requests to vehicles increases. On the other side, we see that the gap decreases with solution values. This was expected: in fact, the relative impact of bad choices made by the greedy algorithm decreases.

2.6.4 Computational results for $V, Q|sd, u, r_i, d_i|CLT$

Tables 2.2 and 2.3 report results for problem $V, Q|sd, u, r_i, d_i|CLT$. The meaning of the column headings is the same as above. "-" indicates that either the instance is infeasible or the greedy algorithm failed in finding a solution. In Table 2.2, three cases are considered for release and due dates: tight-tight (t-t), tight-wide (t-w) and tight release dates without due dates (t). The three remaining cases are reported in Table 2.3.

The main observation from these tables is that the model remains extremely efficient, even when release dates and due dates are considered. Even more, solution

n	V	Q	t-t					t-w					t				
			CPU(s)	feas	greedy	CLT	gapGr	CPU(s)	feas	greedy	CLT	gapGr	CPU(s)	feas	greedy	CLT	gapGr
20	1	1	0.0	3	0	17.3	-	0.0	4	1	17.0	6.7	0.0	5	5	17.4	7.0
20	1	2	0.0	5	0	9.2	-	0.0	5	0	9.2	-	0.0	5	5	9.2	8.7
20	1	4	0.0	5	0	5.0	-	0.0	5	1	4.8	20.0	0.0	5	5	5.0	12.0
20	1	8	0.0	5	0	3.0	-	0.0	5	2	2.8	25.0	0.0	5	5	3.0	26.7
20	2	1	0.0	5	0	8.6	-	0.0	5	0	9.2	-	0.0	5	5	8.6	9.4
20	2	2	0.0	5	2	4.8	22.5	0.0	5	2	4.6	20.0	0.0	5	5	4.6	22.0
20	2	4	0.0	5	0	3.0	-	0.0	5	0	3.2	-	0.0	5	5	3.0	20.0
20	2	8	0.0	5	0	2.0	-	0.0	5	0	2.0	-	0.0	5	5	2.0	50.0
20	3	1	0.0	5	2	6.2	18.3	0.0	4	2	6.5	0.0	0.0	5	5	6.2	10.7
20	3	2	0.0	5	0	3.6	-	0.0	5	2	3.6	12.5	0.0	5	5	3.6	23.3
20	3	4	0.0	5	0	2.8	-	0.0	5	1	2.8	0.0	0.0	5	5	2.8	10.0
20	3	8	0.0	5	0	2.0	-	0.0	5	0	2.0	-	0.0	5	5	2.0	30.0
40	1	1	0.0	5	0	34.2	-	0.0	5	2	31.2	1.7	0.1	5	5	34.0	3.5
40	1	2	0.0	5	0	16.4	-	0.0	5	0	16.6	-	0.0	5	5	16.4	7.4
40	1	4	0.0	5	0	9.2	-	0.0	5	1	8.6	0.0	0.0	5	5	9.0	4.7
40	1	8	0.0	5	0	5.0	-	0.0	5	0	4.8	-	0.0	5	5	5.0	8.0
40	2	1	0.0	5	0	16.6	-	0.0	5	0	16.4	-	0.1	5	5	16.6	3.8
40	2	2	0.0	5	1	9.0	0.0	0.0	5	3	8.8	7.9	0.0	5	5	9.0	2.2
40	2	4	0.0	5	0	4.8	-	0.0	5	0	5.0	-	0.0	5	5	4.8	13.0
40	2	8	0.0	5	0	3.0	-	0.0	5	0	3.0	-	0.0	5	5	3.0	33.3
40	3	1	0.0	5	0	11.0	-	0.0	5	1	11.6	8.3	0.1	5	5	11.0	7.3
40	3	2	0.0	5	0	6.2	-	0.0	5	0	5.8	-	0.1	5	5	6.2	16.2
40	3	4	0.0	5	0	4.0	-	0.0	5	0	4.0	-	0.1	5	5	4.0	0.0
40	3	8	0.0	5	0	3.0	-	0.0	5	0	3.0	-	0.1	5	5	3.0	0.0
80	1	1	0.1	5	1	61.4	3.2	0.1	4	0	62.0	-	0.2	5	5	61.4	1.3
80	1	2	0.0	5	0	31.4	-	0.0	5	0	31.6	-	0.1	5	5	31.4	2.6
80	1	4	0.0	5	1	16.6	6.3	0.0	5	1	15.6	6.3	0.1	5	5	16.6	4.9
80	1	8	0.0	5	0	9.0	-	0.0	5	1	8.8	0.0	0.1	5	5	9.0	8.9
80	2	1	0.1	5	1	31.4	0.0	0.1	5	0	31.0	-	0.3	5	5	31.4	2.6
80	2	2	0.0	5	0	15.8	-	0.0	5	1	16.2	6.3	0.2	5	5	15.8	5.1
80	2	4	0.0	5	2	8.6	6.3	0.0	5	0	8.8	-	0.2	5	5	8.6	7.2
80	2	8	0.0	5	0	5.0	-	0.0	5	0	5.0	-	0.1	5	5	5.0	16.0
80	3	1	0.1	5	1	21.4	4.8	0.2	5	1	21.4	4.5	0.4	5	5	21.4	3.8
80	3	2	0.0	5	1	11.0	0.0	0.0	5	1	11.0	9.1	0.3	5	5	11.0	7.3
80	3	4	0.0	5	0	6.0	-	0.0	5	0	6.0	-	0.2	5	5	6.0	16.7
80	3	8	0.0	5	0	4.0	-	0.0	5	0	4.0	-	0.2	5	5	4.0	15.0
160	1	1	0.6	5	0	120.6	-	0.8	4	0	121.8	-	1.9	5	5	120.6	1.5
160	1	2	0.1	5	1	61.6	1.6	0.1	5	1	61.2	1.6	0.5	5	5	61.6	1.0
160	1	4	0.0	5	1	30.8	3.3	0.1	5	0	31.0	-	0.3	5	5	30.8	2.0
160	1	8	0.0	5	0	16.0	-	0.0	5	2	16.0	3.1	0.3	5	5	16.0	5.0
160	2	1	0.7	5	0	62.2	-	0.7	5	0	61.0	-	2.2	5	5	62.2	1.0
160	2	2	0.1	5	0	31.2	-	0.2	5	1	31.6	0.0	0.9	5	5	31.2	2.6
160	2	4	0.0	5	1	16.2	6.3	0.1	5	0	16.2	-	0.7	5	5	16.2	3.8
160	2	8	0.0	5	0	9.0	-	0.0	5	0	9.0	-	0.7	5	5	9.0	4.4
160	3	1	0.5	5	0	41.6	-	0.6	5	1	40.8	2.5	2.1	5	5	41.6	0.5
160	3	2	0.1	5	1	20.8	4.8	0.2	5	1	21.2	4.8	1.5	5	5	20.8	4.8
160	3	4	0.1	5	0	11.2	-	0.1	5	1	11.0	9.1	1.2	5	5	11.2	5.5
160	3	8	0.0	5	0	6.0	-	0.0	5	0	6.0	-	1.1	5	5	6.0	16.7

Table 2.2: Problem $V, Q|sd, u, r_i, d_i|CLT$ (part 1)

n	V	Q	w-t					w-w					w				
			CPU(s)	feas	greedy	CLT	gapGr	CPU(s)	feas	greedy	CLT	gapGr	CPU(s)	feas	greedy	CLT	gapGr
20	1	1	0.0	5	2	20.6	5.0	0.0	5	1	19.4	5.6	0.0	5	5	20.6	4.9
20	1	2	0.0	5	1	11.2	8.3	0.0	5	1	10.6	0.0	0.0	5	5	10.8	9.3
20	1	4	0.0	5	1	5.8	16.7	0.0	5	1	5.8	16.7	0.0	5	5	5.8	14.0
20	1	8	0.0	5	0	4.0	-	0.0	5	0	4.0	-	0.0	5	5	3.8	11.7
20	2	1	0.0	5	1	10.6	10.0	0.0	5	4	10.8	7.0	0.0	5	5	10.6	4.0
20	2	2	0.0	5	1	6.0	16.7	0.0	5	3	5.8	12.2	0.0	5	5	6.0	10.0
20	2	4	0.0	5	0	4.0	-	0.0	5	0	3.8	-	0.0	5	5	4.0	10.0
20	2	8	0.0	5	0	3.0	-	0.0	5	0	2.6	-	0.0	5	5	3.0	0.0
20	3	1	0.0	5	1	7.8	0.0	0.0	5	1	8.0	0.0	0.0	5	5	7.8	7.9
20	3	2	0.0	5	0	4.4	-	0.0	5	1	4.4	0.0	0.0	5	5	4.4	20.0
20	3	4	0.0	5	0	2.6	-	0.0	5	0	3.0	-	0.0	5	5	2.6	20.0
20	3	8	0.0	5	0	2.0	-	0.0	5	0	2.0	-	0.0	5	5	2.0	50.0
40	1	1	0.0	5	0	39.2	-	0.0	5	0	41.4	-	0.0	5	5	39.2	2.5
40	1	2	0.0	5	0	20.6	-	0.0	5	1	20.8	4.8	0.0	5	5	20.6	4.9
40	1	4	0.0	5	1	11.0	9.1	0.0	5	2	10.8	5.0	0.0	5	5	11.0	3.6
40	1	8	0.0	5	0	6.0	-	0.0	5	1	6.0	16.7	0.0	5	5	6.0	13.3
40	2	1	0.0	5	1	20.8	4.8	0.0	5	1	20.4	4.8	0.1	5	5	20.8	3.9
40	2	2	0.0	5	0	10.6	-	0.0	5	2	10.6	9.5	0.0	5	5	10.6	4.0
40	2	4	0.0	5	2	6.0	8.3	0.0	5	1	6.0	0.0	0.0	5	5	6.0	13.3
40	2	8	0.0	5	0	4.0	-	0.0	5	0	4.0	-	0.0	5	5	4.0	0.0
40	3	1	0.0	5	1	14.6	0.0	0.0	5	1	14.4	0.0	0.1	5	5	14.6	2.8
40	3	2	0.0	5	0	8.0	-	0.0	5	1	7.8	12.5	0.1	5	5	8.0	2.5
40	3	4	0.0	5	0	4.4	-	0.0	5	1	4.8	0.0	0.1	5	5	4.4	10.0
40	3	8	0.0	5	0	3.0	-	0.0	5	0	3.0	-	0.1	5	5	3.0	6.7
80	1	1	0.1	5	2	80.2	0.6	0.1	5	4	80.6	1.2	0.1	5	5	80.2	0.8
80	1	2	0.0	5	1	40.4	0.0	0.0	5	4	41.0	1.2	0.1	5	5	40.4	1.5
80	1	4	0.0	5	0	20.4	-	0.0	5	1	20.8	0.0	0.1	5	5	20.4	4.9
80	1	8	0.0	5	0	10.6	-	0.0	5	1	11.0	9.1	0.1	5	5	10.6	7.6
80	2	1	0.1	5	1	41.0	4.9	0.1	5	2	40.8	1.2	0.3	5	5	41.0	2.0
80	2	2	0.0	5	1	21.0	0.0	0.0	5	2	20.8	2.5	0.2	5	5	21.0	2.9
80	2	4	0.0	5	0	10.8	-	0.0	5	0	11.0	-	0.1	5	5	10.8	7.5
80	2	8	0.0	5	0	6.0	-	0.0	5	1	6.0	16.7	0.1	5	5	6.0	16.7
80	3	1	0.1	5	0	27.6	-	0.1	5	2	28.0	1.8	0.4	5	5	27.6	2.2
80	3	2	0.0	5	1	14.8	0.0	0.0	5	2	14.6	3.6	0.2	5	5	14.8	0.0
80	3	4	0.0	5	1	7.8	12.5	0.0	5	0	8.0	-	0.2	5	5	7.8	12.9
80	3	8	0.0	5	0	5.0	-	0.0	5	0	5.0	-	0.2	5	5	5.0	4.0
160	1	1	0.3	5	1	160.2	0.0	0.3	5	2	160.6	0.9	0.6	5	5	160.2	0.5
160	1	2	0.1	5	0	80.6	-	0.1	5	1	80.8	0.0	0.4	5	5	80.6	0.7
160	1	4	0.0	5	0	40.8	-	0.1	5	3	41.0	1.6	0.3	5	5	40.8	1.0
160	1	8	0.0	5	0	20.8	-	0.0	5	1	21.0	4.8	0.3	5	5	20.8	3.9
160	2	1	0.2	5	0	81.0	-	0.3	5	2	80.2	1.3	1.2	5	5	81.0	0.7
160	2	2	0.1	5	1	41.0	0.0	0.1	5	1	40.8	0.0	0.8	5	5	41.0	1.0
160	2	4	0.0	5	1	21.0	4.8	0.0	5	0	20.4	-	0.8	5	5	21.0	2.9
160	2	8	0.0	5	0	11.0	-	0.0	5	1	11.0	9.1	0.7	5	5	11.0	7.3
160	3	1	0.2	5	0	54.6	-	0.3	5	2	54.0	1.9	1.5	5	5	54.6	0.4
160	3	2	0.1	5	0	27.8	-	0.1	5	2	28.0	1.8	1.3	5	5	27.8	1.5
160	3	4	0.0	5	0	14.8	-	0.1	5	0	15.0	-	1.1	5	5	14.8	1.4
160	3	8	0.0	5	0	8.0	-	0.0	5	0	8.0	-	1.1	5	5	8.0	5.0

Table 2.3: Problem $V, Q|sd, u, r_i, d_i|CLT$ (part 2)

times are much smaller than those reported in Table 2.1, with almost all instances solved in less than one second. This might be due to the fact that release and due dates reduce the solution space. Still, the instances are not trivial to solve, as witnessed by the results related to the greedy algorithm. Indeed, while almost all instances admit feasible solutions, the greedy algorithm fails in finding the optimal solution for most of them. In addition, the greedy algorithm has difficulties even in finding a feasible solution, especially for the case in which release and due dates are tight. This shows that a solution approach smarter than the simple greedy algorithm can provide large advantages.

2.6.5 Computational results for $V, Q|sd, r_i, d_i|CLT$

This section reports results for problem $V, Q|sd, r_i, d_i|CLT$. The case with no release dates and due dates is considered in Table 2.4. Tables 2.5 and 2.6 report results for the case with release and due dates. Contrary to previous results, demands are not forced to be unitary. Optimal solutions are the ones obtained from solving the formulation in Section 2.5.2. Columns are the same as in the former section.

n	V	Q	CPU(s)	feas	greedy	CLT	gapGr(%)	n	V	Q	CPU(s)	feas	greedy	CLT	gapGr(%)
20	1	1	0.1	5	5	13.0	1.5	80	1	1	2.5	5	5	46.4	1.7
20	1	2	0.1	5	5	9.8	8.2	80	1	2	2.9	5	5	35.6	2.8
20	1	4	0.1	5	5	8.8	11.6	80	1	4	15.9	5	5	30.4	9.9
20	1	8	0.1	5	5	8.2	16.9	80	1	8	22.9	5	5	27.6	11.9
20	2	1	0.1	5	5	6.8	2.9	80	2	1	2.5	5	5	23.4	1.6
20	2	2	0.1	5	5	5.4	10.9	80	2	2	3.9	5	5	18.2	5.6
20	2	4	0.1	5	5	5.2	13.0	80	2	4	5.4	5	5	15.6	9.0
20	2	8	0.1	5	5	4.8	13.0	80	2	8	24.4	5	5	14.2	12.8
20	3	1	0.1	5	5	4.8	9.0	80	3	1	3.9	5	5	15.8	3.8
20	3	2	0.1	5	5	3.8	23.3	80	3	2	5.8	5	5	11.8	7.0
20	3	4	0.1	5	5	3.6	26.7	80	3	4	6.0	5	5	10.4	13.5
20	3	8	0.1	5	5	3.2	26.7	80	3	8	20.2	5	5	9.8	16.6
40	1	1	0.3	5	5	25.4	1.6	160	1	1	17.6	5	5	87.4	0.7
40	1	2	0.2	5	5	19.0	3.2	160	1	2	102.9	5	5	65.4	1.2
40	1	4	1.2	5	5	17.0	12.9	160	1	4	227.7	5	5	54.8	10.6
40	1	8	0.5	5	5	14.6	11.0	160	1	8	292.3	5	5	51.6	12.5
40	2	1	0.4	5	5	13.2	1.5	160	2	1	30.8	5	5	44.2	1.8
40	2	2	0.3	5	5	9.8	10.4	160	2	2	29.7	5	5	33.2	3.0
40	2	4	0.4	5	5	8.6	6.9	160	2	4	213.1	5	5	27.4	11.8
40	2	8	0.7	5	5	8.8	11.4	160	2	8	453.7	5	5	25.4	11.0
40	3	1	0.4	5	5	9.0	4.2	160	3	1	39.9	5	5	29.4	2.0
40	3	2	0.4	5	5	7.2	14.0	160	3	2	50.3	5	5	22.2	3.6
40	3	4	0.4	5	5	6.0	14.7	160	3	4	86.4	5	5	19.8	10.1
40	3	8	0.3	5	5	5.0	20.3	160	3	8	162.7	5	5	18.4	12.1

Table 2.4: Problem $V, Q|sd|CLT$

These tables confirm the good-quality of the formulation but also exhibit larger computing times. Largest instances regularly require a few minutes for getting the optimal solution, with or without release dates and due dates. The reason is related to the larger number of variables due to the disaggregated capacity constraints.

n	V	Q	t-t					t-w					t				
			CPU(s)	feas	greedy	CLT	gapGr	CPU(s)	feas	greedy	CLT	gapGr	CPU(s)	feas	greedy	CLT	gapGr
20	1	1	0.0	0	0	-	-	0.0	1	0	13.0	-	0.0	5	5	13.6	7.5
20	1	2	0.0	0	0	-	-	0.0	4	0	11.3	-	0.0	5	5	11.0	13.6
20	1	4	0.0	1	1	9.0	11.1	0.0	4	0	10.0	-	0.0	5	5	10.6	16.9
20	1	8	0.0	2	0	9.0	-	0.0	3	1	10.0	20.0	0.0	5	5	10.0	16.6
20	2	1	0.0	0	0	-	-	0.0	1	0	8.0	-	0.0	5	5	7.0	17.9
20	2	2	0.0	1	0	6.0	-	0.0	4	0	6.0	-	0.0	5	5	5.6	21.5
20	2	4	0.0	2	0	6.0	-	0.0	3	0	5.7	-	0.0	5	5	5.8	24.7
20	2	8	0.0	4	0	5.8	-	0.0	5	0	5.0	-	0.0	5	5	5.4	26.7
20	3	1	0.0	0	0	-	-	0.0	1	0	4.0	-	0.0	5	5	5.2	17.3
20	3	2	0.0	3	0	4.3	-	0.0	5	0	4.6	-	0.0	5	5	4.4	15.0
20	3	4	0.0	2	0	4.0	-	0.0	5	0	4.2	-	0.0	5	5	4.0	32.3
20	3	8	0.0	4	0	4.3	-	0.0	5	1	4.2	0.0	0.0	5	5	4.2	9.0
40	1	1	0.0	0	0	-	-	0.0	1	0	24.0	-	0.2	5	5	26.8	4.8
40	1	2	0.0	1	0	21.0	-	0.1	3	0	20.7	-	0.1	5	5	20.2	9.3
40	1	4	0.0	3	0	18.7	-	0.1	3	0	19.0	-	0.2	5	5	19.2	12.4
40	1	8	0.0	4	0	17.8	-	0.0	4	0	17.8	-	0.1	5	5	18.0	14.5
40	2	1	0.0	0	0	-	-	0.0	2	0	13.0	-	0.2	5	5	13.6	7.6
40	2	2	0.0	1	0	10.0	-	0.1	4	0	10.0	-	0.2	5	5	10.2	13.8
40	2	4	0.0	4	0	9.3	-	0.1	5	0	9.8	-	0.2	5	5	9.4	13.9
40	2	8	0.1	4	0	10.5	-	0.0	5	0	10.0	-	0.2	5	5	10.4	12.3
40	3	1	0.0	0	0	-	-	0.0	4	0	9.0	-	0.3	5	5	9.2	13.2
40	3	2	0.0	3	0	7.3	-	0.0	4	0	8.0	-	0.2	5	5	7.4	16.4
40	3	4	0.0	4	1	6.8	16.7	0.0	4	0	7.0	-	0.1	5	5	6.8	14.9
40	3	8	0.0	4	0	6.5	-	0.0	5	0	6.4	-	0.1	5	5	6.4	22.4
80	1	1	0.0	0	0	-	-	0.2	4	0	47.5	-	1.3	5	5	47.8	4.2
80	1	2	0.4	3	0	39.3	-	1.2	5	0	37.2	-	2.2	5	5	38.2	10.0
80	1	4	0.2	3	0	34.3	-	12.2	5	1	35.2	11.4	2.7	5	5	34.6	11.6
80	1	8	0.4	5	0	32.6	-	0.8	5	0	33.6	-	2.2	5	5	32.2	11.7
80	2	1	0.0	0	0	-	-	0.2	3	0	24.3	-	2.0	5	5	24.0	4.2
80	2	2	0.5	5	0	19.4	-	2.0	5	0	18.8	-	2.4	5	5	18.8	14.9
80	2	4	7.4	5	0	17.2	-	1.2	5	0	17.6	-	9.2	5	5	17.0	11.7
80	2	8	0.2	5	0	17.2	-	0.8	5	0	17.2	-	1.8	5	5	17.2	6.9
80	3	1	0.1	0	0	-	-	0.3	4	0	16.3	-	2.4	5	5	16.2	5.0
80	3	2	1.0	5	0	13.2	-	0.4	5	0	12.4	-	1.4	5	5	13.2	9.1
80	3	4	0.2	5	0	12.0	-	1.7	5	0	12.0	-	1.3	5	5	12.0	13.3
80	3	8	0.5	5	1	12.0	9.1	0.6	5	0	11.6	-	1.6	5	5	11.8	15.3
160	1	1	0.4	0	0	-	-	7.8	5	0	88.4	-	45.6	5	5	88.6	3.8
160	1	2	179.5	5	0	69.4	-	38.5	5	0	69.2	-	346.4	5	5	69.2	6.5
160	1	4	15.3	5	0	64.8	-	22.0	5	0	63.4	-	27.2	5	5	64.6	7.1
160	1	8	18.6	5	0	63.4	-	8.6	5	0	63.4	-	17.4	5	5	63.4	7.0
160	2	1	0.3	0	0	-	-	4.1	5	0	44.8	-	28.5	5	5	44.2	2.2
160	2	2	87.7	5	0	34.4	-	25.9	5	0	34.4	-	44.9	5	5	34.4	6.9
160	2	4	365.6	5	0	32.3	-	9.2	5	0	33.4	-	44.5	5	5	32.0	10.6
160	2	8	12.0	5	0	32.0	-	71.6	5	0	32.2	-	20.5	5	5	31.8	8.8
160	3	1	0.3	0	0	-	-	4.0	5	0	29.6	-	49.8	5	5	29.8	6.0
160	3	2	51.5	5	0	23.2	-	17.2	5	0	23.0	-	26.5	5	5	23.2	9.5
160	3	4	20.4	5	0	22.4	-	24.8	5	0	22.0	-	362.2	5	5	22.2	10.8
160	3	8	92.8	5	0	21.8	-	5.7	5	0	21.8	-	56.4	5	5	21.8	8.3

Table 2.5: Problem $V, Q|sd, r_i, d_i|CLT$ (part 1)

n	V	Q	w-t					w-w					w				
			CPU(s)	feas	greedy	CLT	gapGr	CPU(s)	feas	greedy	CLT	gapGr	CPU(s)	feas	greedy	CLT	gapGr
20	1	1	0.0	0	0	-	-	0.0	2	0	14.0	-	0.0	5	5	14.0	11.4
20	1	2	0.0	2	1	12.0	9.1	0.0	4	0	12.5	-	0.0	5	5	12.2	18.0
20	1	4	0.0	2	0	11.0	-	0.0	1	0	15.0	-	0.0	5	5	11.8	11.9
20	1	8	0.0	2	0	11.5	-	0.0	4	1	11.8	10.0	0.0	5	5	11.4	13.9
20	2	1	0.0	0	0	-	-	0.0	3	0	7.0	-	0.0	5	5	8.4	7.9
20	2	2	0.0	4	0	7.0	-	0.0	4	1	6.5	33.3	0.0	5	5	6.4	15.9
20	2	4	0.0	2	0	7.0	-	0.0	4	1	6.8	16.7	0.0	5	5	6.6	15.4
20	2	8	0.0	5	0	6.4	-	0.0	5	0	6.2	-	0.0	5	5	6.4	12.4
20	3	1	0.0	1	0	5.0	-	0.0	2	0	5.5	-	0.0	5	5	5.6	14.0
20	3	2	0.0	3	0	4.3	-	0.0	5	0	4.6	-	0.0	5	5	4.6	18.0
20	3	4	0.0	4	0	5.0	-	0.0	5	3	4.6	15.0	0.0	5	5	5.0	22.0
20	3	8	0.0	5	1	4.4	25.0	0.0	5	0	4.8	-	0.0	5	5	4.4	23.0
40	1	1	0.0	1	0	26.0	-	0.0	2	0	26.0	-	0.1	5	5	27.0	6.2
40	1	2	0.0	4	0	22.8	-	0.0	5	0	22.6	-	0.1	5	5	21.6	7.4
40	1	4	0.0	4	0	23.5	-	0.0	5	0	23.4	-	0.1	5	5	22.0	10.1
40	1	8	0.0	5	0	21.2	-	0.0	5	0	21.8	-	0.1	5	5	20.8	10.7
40	2	1	0.0	1	0	13.0	-	0.0	3	0	13.7	-	0.2	5	5	14.0	8.8
40	2	2	0.0	4	0	11.5	-	0.0	5	0	11.4	-	0.1	5	5	11.2	10.8
40	2	4	0.0	4	0	11.8	-	0.0	5	0	11.4	-	0.1	5	5	11.6	12.3
40	2	8	0.0	4	1	11.0	10.0	0.0	5	0	11.2	-	0.1	5	5	11.0	14.6
40	3	1	0.0	1	0	9.0	-	0.0	5	0	9.6	-	0.2	5	5	9.0	13.7
40	3	2	0.0	5	0	8.4	-	0.0	5	0	8.4	-	0.1	5	5	8.2	14.7
40	3	4	0.0	5	0	8.2	-	0.0	5	0	8.0	-	0.1	5	5	8.0	12.5
40	3	8	0.0	5	0	7.8	-	0.0	5	0	8.0	-	0.1	5	5	7.8	17.9
80	1	1	0.1	2	0	47.5	-	0.2	4	0	48.8	-	0.8	5	5	48.0	7.5
80	1	2	0.1	5	0	43.6	-	0.3	5	0	42.6	-	0.4	5	5	43.4	6.5
80	1	4	0.1	5	0	41.2	-	0.2	5	0	41.8	-	0.3	5	5	41.2	6.8
80	1	8	0.1	5	0	41.8	-	0.1	5	0	41.2	-	0.3	5	5	41.8	4.7
80	2	1	0.1	4	0	24.0	-	0.3	5	0	24.8	-	1.3	5	5	23.8	7.5
80	2	2	0.1	5	0	22.2	-	0.2	5	1	22.0	4.5	0.7	5	5	22.2	5.7
80	2	4	0.1	5	1	21.4	4.5	0.1	5	0	21.4	-	0.4	5	5	21.2	5.7
80	2	8	0.1	5	0	21.4	-	0.1	5	0	21.4	-	0.4	5	5	21.4	5.6
80	3	1	0.1	2	0	16.5	-	0.2	5	0	17.0	-	1.4	5	5	16.8	8.4
80	3	2	0.1	5	0	15.0	-	0.1	5	0	14.8	-	0.5	5	5	15.0	9.3
80	3	4	0.1	5	0	15.0	-	0.1	5	0	15.2	-	0.4	5	5	15.0	5.3
80	3	8	0.1	5	0	15.0	-	0.1	5	0	14.8	-	0.5	5	5	15.0	4.0
160	1	1	2.7	5	0	91.4	-	4.5	5	0	91.2	-	17.0	5	5	91.2	6.4
160	1	2	1.7	5	0	83.6	-	3.6	5	0	83.6	-	5.0	5	5	83.6	3.3
160	1	4	0.4	5	0	81.6	-	0.5	5	0	81.4	-	1.7	5	5	81.6	2.5
160	1	8	0.3	5	0	82.0	-	0.4	5	0	81.4	-	1.4	5	5	81.8	1.7
160	2	1	1.7	5	0	46.4	-	3.5	5	0	45.8	-	12.3	5	5	46.4	5.2
160	2	2	0.8	5	0	42.4	-	2.1	5	0	41.8	-	2.6	5	5	42.2	4.3
160	2	4	0.5	5	0	42.0	-	0.9	5	0	41.8	-	2.4	5	5	42.0	3.3
160	2	8	0.4	5	0	40.8	-	0.6	5	1	41.2	2.4	3.2	5	5	40.8	3.9
160	3	1	1.7	5	0	30.2	-	2.8	5	0	30.8	-	21.1	5	5	30.0	9.3
160	3	2	0.6	5	0	27.6	-	1.2	5	0	28.6	-	4.7	5	5	27.6	5.8
160	3	4	1.0	5	0	28.2	-	0.4	5	0	28.4	-	3.3	5	5	28.2	1.4
160	3	8	0.3	5	0	27.6	-	0.4	5	1	27.8	3.7	3.4	5	5	27.6	2.9

Table 2.6: Problem $V, Q|sd, r_i, d_i|CLT$ (part 2)

These instances are also specially difficult for the greedy algorithm. In fact, it fails in finding a feasible solution for most of the instances and shows larger gaps compared to instances with unitary demands, when feasible solution are found. The additional complexity implied by the packing of non-unitary requests makes the greedy algorithm not effective.

2.6.6 Summary of results

Figure 2.8 presents aggregated data from tables 2.1-2.6. The nine graphs are organized in a 3×3 grid. The graphs report the average values of CPU, CLT and $gapGr$ over instances with the same value of n, V and Q, respectively. Each graph contains six lines, one for each problem in tables 2.1-2.6.

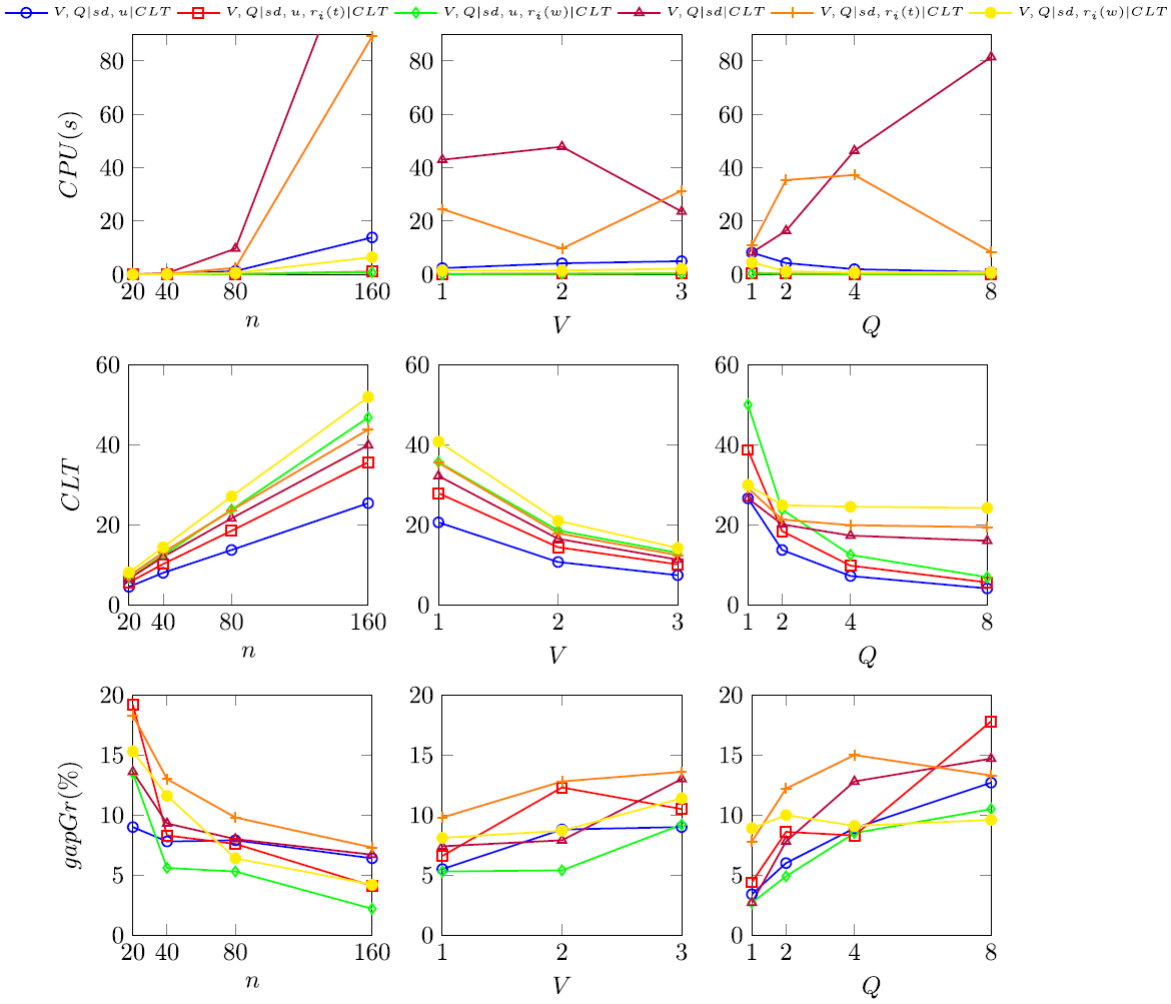


Figure 2.8: Average CPU, CLT and $gapGr$ as functions of n, V and Q

The CPU graphs clearly show that the hardest problem instances are $V,Q|sd|CLT$ and $V,Q|sd,r_i|CLT$ with tight release dates, where the CPU depends strongly on the value of n. For problem $V,Q|sd|CLT$, Q is also strongly correlated to computing time.

As for the value of the objective function CLT, all problems show similar behaviour: it grows with n and decreases with V and Q. As expected, instances of $V,Q|sd,r_i|CLT$ with wide release dates required the highest CLT.

From the last 3 graphs it is possible to see that the gap between the optimal value and the value of the solution of the greedy algorithm is the highest for problem $V,Q|sd,r_i|CLT$ with tight release dates, and the smallest for $V,Q|sd,u,r_i|CLT$ with wide release dates. For all problems, the gap decreases with n and increases with Q. If the gap is plotted as a function of V it slightly grows for four problems and it fluctuates for two of them.

2.6.7 Additional experiments on larger instances

In these experiments the computational limits of formulation (2.9)–(2.14) are evaluated. Additional tests are performed by starting from the instances of problem $V,Q|sd|CLT$ because they were the hardest to solve (in terms of CPU time). The configuration of the parameters with the biggest values was considered and 10 additional instances were generated. 5 instances were generated by increasing all parameters by 50%, and 5 instances were generated by increasing all parameters by 50% except the number of requests n that was kept to 160. The resulting number of stations was $m = 15$. Tables 2.7 and 2.8 report the results of these new tests. In these tables, value $gapLR$ gives the gap between the value of the linear relaxation at the root node (LR) and the upper bound on the solution value. It is calculated as $\frac{LR-CLT}{CLT} \times 100$. Column $gapGrOPT$ gives the gap between the optimal solution value and the greedy solution value, while column $gapGrUB$ gives the gap between the upper bound value and the greedy solution value. They are calculated as $\frac{greedy-CLT}{CLT} \times 100$, where $greedy$ is the greedy solution value. Column opt reports reports the number of instances solved to optimality. The remaining column headings have the same meaning as in previous tables.

m	n	V	Q	CPU(s)	feas	greedy	opt	CLT	gapGrOPT	gapGrUB	gapLR
15	240	5	12	3363.2	5	5	1	15.0	20.0	8.9	-45.5

Table 2.7: Problem $V,Q|sd|CLT$

m	n	V	Q	CPU(s)	feas	greedy	opt	CLT	gapGrOPT	gapGrUB	gapLR
15	160	5	12	275.0	5	5	5	10.6	17.1	0.0	-39.6

Table 2.8: Problem $V, Q|sd|CLT$

The results show that when $n = 240$ only 1 out of 5 instances was solved to optimality. Instead, when $n = 160$, all instances were solved to optimality. This proves that the only parameter that has an impact on the time taken to solve the model is the number of requests n, and that all other parameters (including the number of stations m) have little or no impact.

2.7 Conclusions and perspectives

In this chapter a new class of Pickup and Delivery problems where the stations are located on rings was introduced. A classification scheme was proposed. It was investigated the complexity of the variants in which the vehicles are allowed to move in a single direction and the objective is the minimization of the maximum number of tours. A polynomial-time algorithm for some variants was described and the NP-hardness of the remaining variants was proved.

For the NP-hard variants, ILP formulations were proposed and computational tests were performed to evaluate these formulations. Experiments on a large number of instance show the impressive efficiency the proposed formulations. All instances, with up to 160 requests, could be solved in a few minutes. Comparisons with a simple and practically relevant greedy algorithm also confirmed the intrinsic difficulty of the problems/instances and the usefulness of applying exact solution schemes.

Different future research directions are possible. In parallel to this work, other variants of these problems were addresses, e.g., problems where vehicles are allowed to move in different directions or problems with alternative objective functions. Another interesting direction would be to consider different network topologies such as lines or other geometric shapes that can be encountered in practice. Also, autonomous vehicles are bound to use electric engines. A future step of our research should be to investigate the issues implied by the limited autonomy of electric vehicles (range anxiety, recharging policies...). The proposed classification scheme could be enriched to account for new variants where the problem of energy management is considered.

Note finally, that the transportation services investigated in this chapter can also be interpreted as services applying on-demand transportation (the booking part) on fixed lines, with the particularity that the first and last stations are identical (the ring). The chapter then shows the advantages of the booking system compared to the usual fixed lines system in which every user is allowed to enter a vehicle as long

as a seat is available. This new angle also offers interesting perspectives.

Chapter 3

Complexity of problem $V, 1|sd, u|CLT$

This chapter contains the proof of NP-hardness of problem $V, 1|sd, u|CLT$, introduced in Chapter 2, as the proof is too long.

The proof is in two steps:

1. The Eulerian Path Partition Problem (EPP) is introduced and proven NP-complete. This is done through a polynomial-time reduction from the well-known 3-SAT problem, which is NP-complete.
2. A polynomial-time reduction from EPP to the decision version of $V, 1|sd, u|CLT$ is then proposed. Given that EPP is NP-complete, this proves that $V, 1|sd, u|CLT$ is NP-hard.

The EPP is defined as follows.

Definition 9. *Let us consider a directed acyclic graph $G = (X, E)$, together with 2 nodes $s \in X$ and $p \in X$ and 2 integer numbers K and T. We suppose that, for any node $x \in X$, a path from s to x and a path from x to p exist in G. The EPP consists in determining if a partition of E into K arc-disjoint paths can be found, with exactly T arcs in each path. In this case, we say that G is $(K, T) - EPP$.*

3.1 Reduction of 3-SAT to EPP

Definition 10. *Let us consider a collection C of clauses on a finite set U of Boolean variables where every clause in C is made up by exactly three literals. 3-SAT is the problem of deciding whether a truth assignment for U exists, i.e. , an assignment that satisfies all clauses in C.*

We will follow an approach very similar to the construction procedure which was proposed in Itai et al., 1982. We consider a 3-SAT instance $Z = \{z_1, ..., z_N\}$ (variable set) and $C = \{c_1, ..., c_S\}$ (3-clause set), with N variables and S clauses. We suppose that for any $j = 1, ..., N$, the number of occurrences of z_j in C is equal to the number of occurrences of $\neg z_j$ and we denote it by $u(j)$. We call this assumption the *Well-Balanced Hypothesis*. We know from Lemma 3.1 (Itai et al., 1982) that the resulting restriction of 3-SAT remains NP-Complete. We set $U = \sum_{j=1}^{N} u(j)$ and see that the total number of literals in the clauses is $3S = 2U$. In what follows, we assume that each occurrence of z_j and $\neg z_j$ is associated with a number in $\{1, \ldots, u(j)\}$.

We build a graph $H = (X, E)$ such that the 3-SAT instance is feasible if and only if a graph which is slightly modified with respect to H is $(3S + 2U, 11)$-EPP. H is acyclic and is composed of six layers. The construction is illustrated on Figures 3.1 and 3.2 for an instance with 3 variables and 2 clauses $c_1 = (z_1 \vee \neg z_2 \vee z_3)$, $c_2 = (\neg z_1 \vee z_2 \vee \neg z_3)$. Node set X is given by Table 3.1. Each line defines a given category of nodes, at a given layer, and introduces both a name and a notation for the nodes of the category. The total number of nodes in each category is reported in the last column. All nodes are visible in Figure 3.1. One can notice that layers 3 and 4 contain exactly $2U$ nodes, that is, the number of literals in the clauses.

Layer	Category	Symbol		Number
1	Source node	s		1
2	Clause nodes	$c_1, ..., c_S$		S
	Low layer variable nodes	(j, u)	$(j \in \{1, \ldots, N\}, u \in \{1, \ldots, u(j)\})$	U
3	First middle nodes	(j, u, ϵ)	$(j \in \{1, \ldots, N\}, u \in \{1, \ldots, u(j)\}, \epsilon \in \{0, 1\})$	$2U$
4	Second middle nodes	$(j, u, \epsilon)^*$	$(j \in \{1, \ldots, N\}, u \in \{1, \ldots, u(j)\}, \epsilon \in \{0, 1\})$	$2U$
5	Bottleneck node	Q		1
	Top layer variable nodes	$(j, u)^*$	$(j \in \{1, \ldots, N\}, u \in \{1, \ldots, u(j)\})$	U
6	Sink node	p		1

Table 3.1: Node set V

Arc set E includes clause-related and variable-related arcs. Figure 3.1 reports all these arcs for our illustrative example.

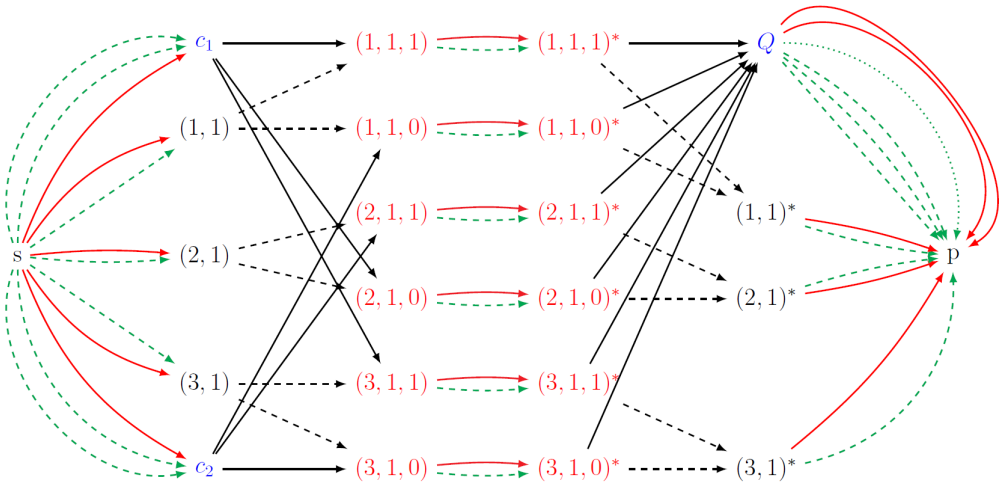


Figure 3.1: A Graph $H = (X, E)$, with 2 clauses c_1, c_2 and 3 variables z_1, z_2, z_3

Clause-related arcs are defined in Table 3.2 and are shown in figures 3.1 and 3.2, with information on arc lengths on the latter. The structure of the table is similar to that of Table 3.1, with an additional column for arc lengths. Given a clause c_k and a literal y_i in this clause, notation $lit(c_k, i)$ in the table indicates a triplet composed of: the index of the associated variable, the occurrence number of the literal, 0 or 1 if the variable is in its negative or positive form in the literal, respectively. For example, the three literals associated with clause $c_1 = (z_1 \vee \neg z_2 \vee z_3)$ are $lit(c_1, 1) = (1, 1, 1)$, $lit(c_1, 2) = (2, 1, 0)$, and $lit(c_1, 3) = (3, 1, 1)$. Consequently, the three c-variable arcs for the clause are arcs $(c_1, (1, 1, 1))$, $(c_1, (2, 1, 0))$ and $(c_1, (3, 1, 1))$. Note that first middle nodes have exactly one in-going clause-related arc each. Note also that the notation introduced in the table does not always enable to distinguish between parallel arcs (c-default arcs, good bottleneck arcs, bad bottleneck arcs are not distinguished). This notation is kept to ease readability. Clause-related arcs represent exactly $3S$ arcs between every successive layers except between layers 3 and 4 where no clause-related arcs are introduced (remembering that $3S = 2U$).

Layers	Category	Symbol		Length	Number
$(1,2)$	c-id arcs	$(s,c_k)^{Id}$	$(k \in \{1,\ldots,S\})$	5	S
	first c-def arcs	$(s,c_k)^{Def}$	$(k \in \{1,\ldots,S\})$	1	S
	second c-def arcs	$(s,c_k)^{Def}$	$(k \in \{1,\ldots,S\})$	1	S
$(2,3)$	first c-variable arcs	$(c_k,(j,u,\epsilon))$	$(k \in \{1,\ldots,S\},\ (j,u,\epsilon) = lit(c_k,1))$	1	S
	second c-variable arcs	$(c_k,(j,u,\epsilon))$	$(k \in \{1,\ldots,S\},\ (j,u,\epsilon) = lit(c_k,2))$	1	S
	third c-variable arcs	$(c_k,(j,u,\epsilon))$	$(k \in \{1,\ldots,S\},\ (j,u,\epsilon) = lit(c_k,3))$	1	S
$(4,5)$	bottleneck arcs	$((j,u,\epsilon)^*,Q)$	$(j \in \{1,\ldots,N\}, u \in \{1,\ldots,u(j)\}, \epsilon \in \{0,1\})$	1	$2U$
$(5,6)$	good bottleneck arcs	$(Q,p)^{good}$	S copies	3	S
	bad bottleneck arcs	$(Q,p)^{bad,Id}$	$U-S$ copies	7	$U-S$
	bad bottleneck arcs	$(Q,p)^{bad,Def}$	$3S-U$ copies	6	$3S-U$

Table 3.2: Clause-related arcs in set E

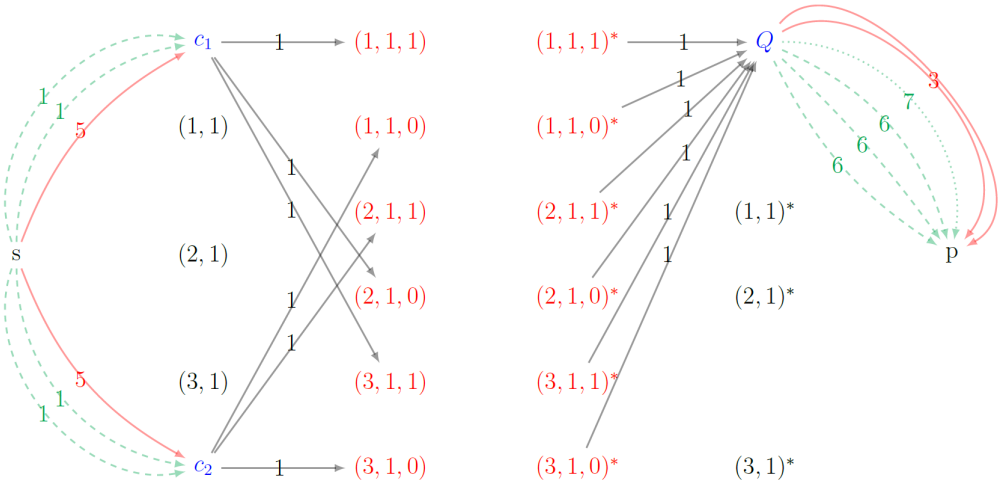


Figure 3.2: Arc lengths for clause-related arcs of graph $H = (X, E)$ of figure 3.1

Variable-related arcs are defined in Table 3.3. This table reads as Table 3.2. Arcs can be seen on Figure 3.1 for our example. Top-second arcs induce what we call a *Saw Pattern*. This pattern cannot clearly be observed on Figure 3.1 because $u(j) = 1$ for every variable z_j. The pattern is illustrated on figure 3.3, with a variable having three occurrences in the clauses. In the definition of these arcs, notation $u + \epsilon$ is assumed to give value 1 when $u = u(j)$ and $\epsilon = 1$. The Saw Pattern is needed to have consistency among different occurrences of the same variable. As shown in Figure 3.1, it defines two complementary perfect matchings between the three *second middle nodes* having the same value for ϵ and the three *top layer variable nodes*. Similarly to clause-related arcs, variable-related arcs involve exactly $2U$ arcs (that is, $3S$) between every successive layers except between layers 3 and 4, with $4U$ arcs. Considering the two sets of arcs, the graph contains exactly $4U$ arcs between every successive layers.

Layers	Category	Symbol		Length	Number
$(1,2)$	low-first-id arcs	$(s,(j,u))^{Id}$	$(j \in \{1,\dots,N\}, u \in \{1,\dots,u(j)\})$	7	U
	low-first-def arcs	$(s,(j,u))^{Def}$	$(j \in \{1,\dots,N\}, u \in \{1,\dots,u(j)\})$	3	U
$(2,3)$	low-second arcs	$((j,u),(j,u,\epsilon))$	$(j \in \{1,\dots,N\}, u \in \{1,\dots,u(j)\}, \epsilon \in \{0,1\})$	1	$2U$
$(3,4)$	middle-id arcs	$((j,u,\epsilon),(j,u,\epsilon)^*)^{Id}$	$(j \in \{1,\dots,N\}, u \in \{1,\dots,u(j)\}, \epsilon \in \{0,1\})$	1	$2U$
	middle-def arcs	$((j,u,\epsilon),(j,u,\epsilon)^*)^{Def}$	$(j \in \{1,\dots,N\}, u \in \{1,\dots,u(j)\}, \epsilon \in \{0,1\})$	2	$2U$
$(4,5)$	top-second	$((j,u,\epsilon)^*,(j,u\ \epsilon)^*)$	$+ (j \in \{1,\dots,N\}, u \in \{1,\dots,u(j)\}, \epsilon \in \{0,1\})$	1	$2U$
$(5,6)$	top-first-id arcs	$((j,u)^*,p)^{Id}$	$(j \in \{1,\dots,N\}, u \in \{1,\dots,u(j)\})$	1	U
	top-first-def arcs	$((j,u)^*,p)^{Def}$	$(j \in \{1,\dots,N\}, u \in \{1,\dots,u(j)\})$	4	U

Table 3.3: Variable-related arcs in set E

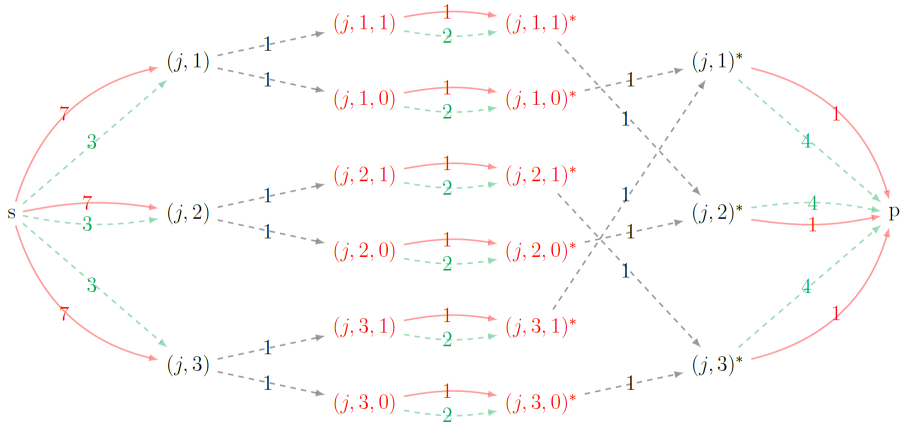


Figure 3.3: Variable-related arcs for a given variable z_j with $u(j) = 3$, with the Saw Pattern between layers 4 and 5.

From graph $H = (X, E)$, we define a second graph $H^* = (X^*, E^*)$ by replacing every arc of length $h > 1$ with a chain of h arcs of length 1. This new graph is useful to meet the definition of EPP in which the length of a path is given by its number of arcs. Then, finding a partition of E^* in K arc-disjoint paths with T arcs in each path is equivalent to finding a partition of E in K arc-disjoint paths with length T. Note that arc lengths are not completely arbitrary. In fact, it is possible to use infinite combinations of arc lengths. However, their values have to be chosen in such a way that set E^* can be partitioned into a set of arc-disjoint paths.

We now see that 3-SAT reduces to EPP. We first see how to build a partition of E^* into a set of arc-disjoint paths starting from a valid truth assignment of variables in Z. Then we see how to build a valid assignment starting from an arc-disjoint partition of the arcs of H^*.

Theorem 6. *3-SAT instance* (Z, C) *is feasible iff* $H^* = (X^*, E^*)$ *is* $(3S + 2U, 11)$-*EPP.*

Proof. Let us first assume that our 3-SAT instance is positive, that is, it admits a feasible solution $z = (\epsilon_1, \ldots, \epsilon_N)$. We see how the arcs of H^* can be partitioned into $3S + 2U$ paths with 11 arcs or equivalently the arcs of H into $3S + 2U$ paths with length 11. The construction is illustrated in Figures 3.4 and 3.5, pursuing with the same two clauses c_1 and c_2, and considering assignment $z = \{1, 1, 1\}$. The partition is as follows:

- It is first generated a path for every occurrence u of literal z_j and a path for every occurrence u of literal $\neg z_j$ in the clauses, for a total of $2 \times U$ paths:
  - **(j,u)-identifier** path: $s \xrightarrow{Id} (j, u) \rightarrow (j, u, \neg\epsilon_j) \xrightarrow{Id} (j, u, \neg\epsilon_j)^* \rightarrow (j, u + (\neg\epsilon_j))^* \xrightarrow{Id} p$; (the red paths in fig. 3.4)
  - **(j,u)-default** path: $s \xrightarrow{Def} (j, u) \rightarrow (j, u, \epsilon_j) \xrightarrow{Def} (j, u, \epsilon_j)^* \rightarrow (j, u + \epsilon_j)^* \xrightarrow{Def} p$; (the green paths in fig. 3.4)

  These paths cover all variable-related arcs, except middle-id arcs $((j, u, \epsilon_j), (j, u, \epsilon_j)^*)^{Id}$ and middle-def arcs $((j, u, \neg\epsilon_j), (j, u, \neg\epsilon_j)^*)^{Def}$. They all have a length equal to 11. Figure 3.4 shows these paths.

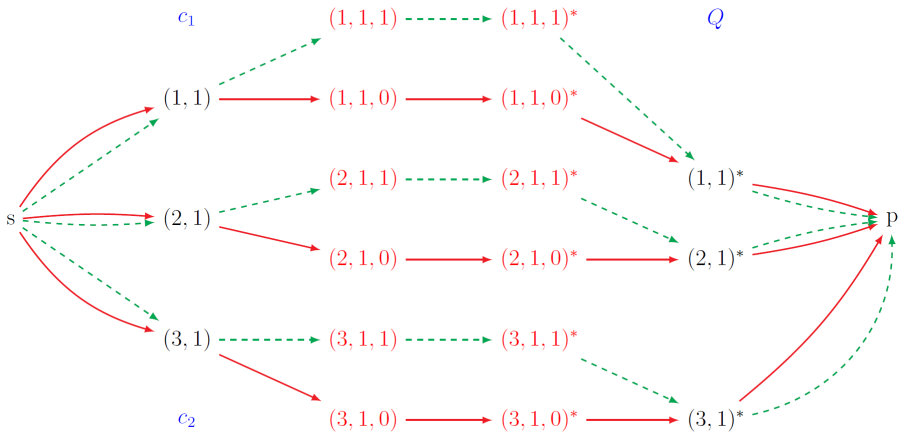


Figure 3.4: $(j, u)-$identifier and $(j, u)-$default paths in graph H

- Then, three paths are generated for every clause c_k, for a total of $3 \times S$ paths. A clause involves three literals that can be matched to a variable z_j and an occurrence u. Each path visits, in sequence, nodes s, c_k, (j, u, ϵ), $(j, u, \epsilon)^*$, Q and p, where $\epsilon = 1$ if the literal is z_j and $\epsilon = 0$ if the literal is $\neg z_j$ (see

definition of clause-variables arcs). The arcs traversed depend on the type of path as follows.

- **c-identifier** path: among the three literals, at least one confirms the validity of the clause, i.e., its assignment is such that the clause is satisfied; we arbitrarily choose one such literal (in case there is more than one literal that satisfies the clause) and we generate path $s \xrightarrow{Id} c_k \rightarrow (j,u,\epsilon) \xrightarrow{Id} (j,u,\epsilon)^* \rightarrow Q \xrightarrow{Good} p$; (in red in figure 3.5)
- two **c-default** paths: a path is generated for every of the two remaining literals; the literals can satisfy the clause or not; the path is $s \xrightarrow{Def} c_k \rightarrow (j,u,\epsilon) \xrightarrow{Id} (j,u,\epsilon)^* \rightarrow Q \xrightarrow{Bad,Id} p$ if the literal satisfies the clause, $s \xrightarrow{Def} c_k \rightarrow (j,u,\epsilon) \xrightarrow{Def} (j,u,\epsilon)^* \rightarrow Q \xrightarrow{Bad,Def} p$ otherwise (with the same definition as above for ϵ). (in green in figure 3.5)

Figure 3.5 shows the $c-$identifier and the $c-$default paths.

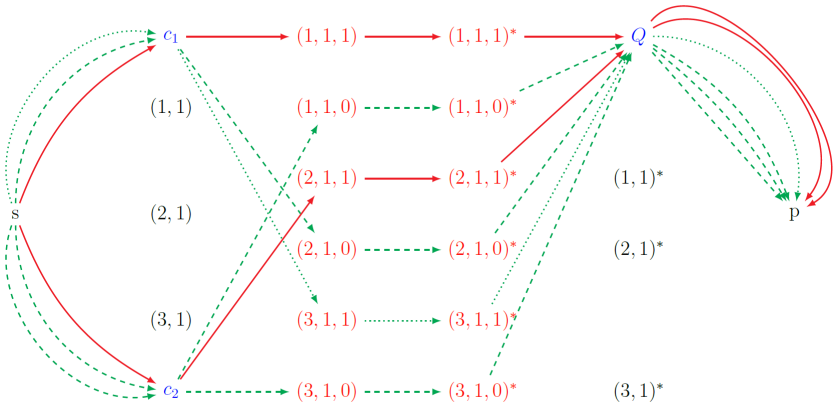


Figure 3.5: $c-$identifier and $c-$default paths in graph H

All paths have a length equal to 11. They cover all remaining middle-id and middle-def arcs. Indeed, an arc $((j,u,\epsilon),(j,u,\epsilon)^*)^{Id}$ is traversed when the literal is true in the clause, that is, when $\epsilon = \epsilon_j$. They also cover all clause-related arcs. Especially, for every clause, the c-id and the two c-def arcs are covered, as well as one good bottleneck arc and two bad bottleneck arcs. Furthermore, thanks to the well-balanced assumption, exactly U literals satisfy their clause and U literals do not. So, U literals cover the U bad bottlenecks arcs with cost 6, the other bottleneck arcs being covered by the other literals (S for the good bottleneck arcs, U-S for the remaining bad bottleneck arcs).

All together, the $3S + 2U$ paths partition E in paths of length 11, which proves that H^* is $(3S + 2U, 11)$-EPP.

Conversely, let us now assume that H^* is $(3S + 2U, 11)$-EPP. We will see that 3-SAT is positive. Indeed, if H^* is $(3S + 2U, 11)$-EPP, it means that there exists a partition of the arcs of H into $M = 3S + 2U$ paths with length 11, which we denote by $\Gamma_1, ..., \Gamma_M$. Then we see that those paths, which all contain one first middle node (j, u, ϵ), can be split into 4 categories:

1. Those who reach node (j, u, ϵ) with a sub-path $s \xrightarrow{Id} (j, u) \rightarrow (j, u, \epsilon)$ of length 8. To have a length 11, they must have the form of **(j,u)-identifier** paths. Every such path contains arc $((j, u, \epsilon), (j, u, \epsilon)^*)^{Id}$.

2. Those who reach node (j, u, ϵ) with a sub-path $s \xrightarrow{Def} (j, u) \rightarrow (j, u, \epsilon)$ of length 4. They must be **(j,u)-default** paths. Every such path contains arc $((j, u, \epsilon), (j, u, \epsilon)^*)^{Def}$.

3. Those who reach node (j, u, ϵ) with a sub-path $s \xrightarrow{Id} c_k \rightarrow (j, u, \epsilon)$ of length 6. They must finish with **good bottleneck** arcs and be **c-identifier** paths.

4. Those who reach node (j, u, ϵ) with a sub-path $s \xrightarrow{Def} c_k \rightarrow (j, u, \epsilon)$ of length 2. They should finish with **bad bottleneck** arcs. They are **c-default** paths. Furthermore, $U - S$ of them contain bad bottleneck arcs of size 7, and, so, also contain an arc $((j, u, \epsilon), (j, u, \epsilon)^*)^{Id}$.

Every node (j, u, ϵ) is traversed by two paths:

- One path Γ_m which is either a **(j,u)-identifier** path or a **(j,u)-default** path. We call this path the **representative path** of (j, u, ϵ). We see that if the **representative path** of (j, u, ϵ) is a **(j,u)-default** path then the **representative path** of $(j, u, \neg\epsilon)$ is a **(j,u)-identifier** path and vice-versa. Indeed, one path starts with sequence $s \xrightarrow{Id} (j, u)$, the other with sequence $s \xrightarrow{Def} (j, u)$.

- One path Γ_{m^*} which is either a **c-default** or a **c-identifier** path for some clause c_k.

We can now derive from paths $\Gamma_1, ..., \Gamma_M$, an assignment of $\{0, 1\}$ values to variables z_j. For every clause c_k, we consider the node (j, u, ϵ) traversed by the c-identifier path. We give value ϵ to variable z_j. After this first step, it is possible that not all variables are assigned. We complete with random values for other variables. These values are proven consistent, *i.e.*, the resulting assignment makes 3-SAT positive.

We use the following property of the Saw Pattern that we call the Saw Pattern Property. Given a variable z_j, we know that paths $\Gamma_1, ..., \Gamma_M$ contain $u(j)$

(j,u)-default paths and that each of these (j,u)-default paths reaches one of the $u(j)$ nodes $(j,u)^*$. The Saw Pattern imposes that the latter are reached from nodes (j,u,ϵ_u) with all ϵ_u equal (using the arcs of one of the two perfect-matchings that compose the Saw Pattern). This also implies that, if variable z_j gets value ϵ, then all arcs $((j,u,\neg\epsilon),(j,u,\neg\epsilon)^*)^{Id}$ have to be in (j,u)-identifier paths and all arcs $((j,u,\epsilon),(j,u,\epsilon)^*)^{Def}$ have to be in (j,u)-default paths. For example, in Figure 3.3, the first occurrence of variable z_j is negative (*i.e.* takes value 0), if and only if node $(j,1,1)$ is reached with a (j,u)-identifier path, that must therefore finish with arc $((j,2)^*,p)$ of length 1. It means that arc $((j,2)^*,p)$ of length 4 must be in a (j,u)-default path together with arc $((j,2,0),(j,2,0)^*)^{Def}$, forcing also arc $((j,2,1),(j,2,1)^*)^{Id}$ to be in a (j,u)-identifier path and thus forcing the second occurrence of variable j to get value 0. The same happens for the third occurrence.

Assume now that a variable z_j receives two values ϵ and ϵ' from two clauses c and c'. It means that arcs $((j,u,\epsilon),(j,u,\epsilon)^*)^{Id}$ and $((j,u',\epsilon'),(j,u',\epsilon')^*)^{Id}$ are in the c-identifier path and the c'-identifier path, respectively. It implies that arcs $((j,u,\epsilon),(j,u,\epsilon)^*)^{Def}$ and $((j,u',\epsilon'),(j,u',\epsilon')^*)^{Def}$ are in the (j,u)-default path and (j,u')-default paths, respectively. Equality $\epsilon=\epsilon'$ follows from the Saw Pattern Property.

By construction, we also know that all clauses are satisfied for these values, which concludes the proof.

□

3.2 Reduction of EPP to $V,1|sd,u|CLT$

In this section we see that EPP can be reduced to $V,1|sd,u|CLT$.

Theorem 7. *EPP can be reduced to* $V,1|sd,u|CLT$.

Proof. Let $\mathcal{I}_{EPP}=(G(X,E),s,p,K,T)$ be a non trivial instance of EPP, i.e., the problem of determining whether there exists K disjoint paths of length T from s to t in G. Non trivial means that G has the following properties:

1. $d^-(x)=d^+(x)$ for any node $x\neq s,p$, where $d^-(x)$ denotes the *in-degree* of x and $d^+(x)$ denotes its *out-degree*
2. $d^+(s)=K$
3. $|E|=KT$

This implies that $d^+(s)=d^-(p)$. It is easy to see that any instance that does not meet these requirements is trivially not $(K,T)-EPP$. Note that it is enough to

consider non-trivial instances in the reduction because if non-trivial instances could be solved with the reduction then all instances would be solved.

We build an instance $\mathcal{I}$ of $V, 1|sd, u|CLT$ as follows. We introduce a ring with $m = |X| + 1$ stations. We first define station 0, the depot. The other stations have a one-to-one correspondence with the vertices in X as follows. We first sort the nodes in X in a topological order (we can do that because G is a directed acyclic graph) and reverse this order. Node p is then the first node and node s the last. We then associate one station with each node following this order, thus having station 1 representing node p and station $|X|$ representing node s. We denote $stat(x)$ the station associated with node $x \in X$. We define unitary distances between successive stations, which gives a tour length $L = |X| + 1$. For every arc $(x, y) \in E$, we introduce a request in $\mathcal{R}$ defined by $s_i = stat(x)$, $t_i = stat(y)$. This request is denoted $r(x, y)$. Since we considered nodes in a reversed topological order, all requests cover the depot. We complete $\mathcal{R}$ by adding K requests $(0, stat(s))$ and K requests $(stat(p), 0)$. This way, $|\mathcal{R}| = KT + 2K$. We finally set $V = K$ and $CLT = L \times (T + 1)$. Note that this construction is polynomial.

This construction is illustrated with a simple example. We consider the graph G of figure 3.6.

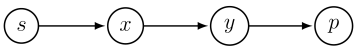
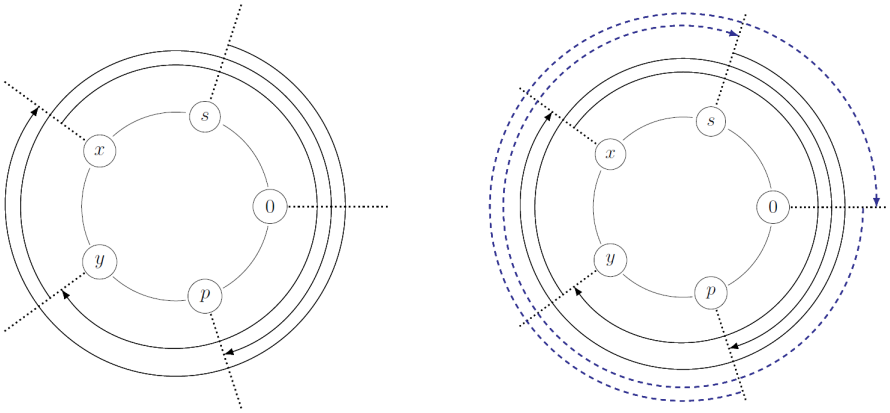


Figure 3.6: Graph G

The resulting ring and set of requests are shown in Figure 3.7.



(a) Ring and requests from arcs in E

(b) Ring and complete set of requests

Figure 3.7: Ring and requests constructed from G

Assume first that $\mathcal{I}_{EPP}$ is positive. This means that set E can be partitioned into K arc-disjoint paths $\mathcal{P}_k$, each of length T. We build a feasible schedule for $\mathcal{I}$ in the following way. Every vehicle k ($1 \leq k \leq V$) starts from the depot by serving request $(0, stat(s))$. Then, it serves all requests $r(x, y)$ for $(x, y) \in \mathcal{P}_k$, in the order defined by this path. It finally serves request $(stat(p), 0)$. This sequence starts from the depot, finishes at the depot and is such that the ending station of every request is the starting station of the next request. It thus defines a feasible schedule without any intermediate empty traveling between requests. The schedule traverses exactly T times the depot, once for every request $r(x, y)$ with $(x, y) \in \mathcal{P}_k$. So, the number of tours is $T + 1$. All $KT + 2K$ requests are covered by the V vehicles, which proves that $\mathcal{I}$ is positive.

Assume now that $\mathcal{I}$ is feasible. This means that there exists a schedule with the $K(T + 2)$ requests assigned to the K vehicles, each vehicle making $T + 1$ tours. We first see that vehicles never travel empty in this schedule.

Segment $[0, 1]$ of the ring is covered by the KT requests $r(x, y)$ with $(x, y) \in E$ and K requests $(0, stat(s))$, that is, $K(T + 1)$ requests. We show that all other segments are covered by the same number of requests. We can easily check that at every station $stat(x)$ exactly K requests stop, either because $d^-(x) = K$ or because $x = s$ and the K requests $(0, stat(s))$ stop (remember that $d^-(s) = 0$ so no other request stops). Similarly, exactly K requests start from every station $stat(x)$, either because $d^+(x) = K$ or because $x = p$ and the K requests $(stat(p), 0)$ start. This shows that every segment is covered by exactly $K(T + 1)$ requests which, in turn, means that none of these segments is traveled empty in a schedule composed of $K(T + 1)$ tours.

We now consider the schedule of a given vehicle k. It executes $T + 1$ tours so it covers at most T requests $r(x, y)$ with $(x, y) \in E$. As the KT requests $r(x, y)$ with $(x, y) \in E$ are shared by the K vehicles, every vehicle exactly fulfills T of these requests. Also, every vehicle starts and ends at the depot, and given that the vehicle never travels empty, this means that it starts with a request $(0, stat(s))$ and ends with a request $(stat(p), 0)$. Given the total number K of $(0, stat(s))$ requests and $(stat(p), 0)$ requests, each vehicle exactly fulfills one request $(0, stat(s))$ and one request $(stat(p), 0)$.

One can conclude that every vehicle starts from the depot with a request $(0, stat(s))$, continues with T requests $r(x, y)$ with $(x, y) \in E$ and finishes with request $(stat(p), 0)$, without any segment traveled empty in between. It also means that the arcs $(x, y) \in E$ associated with the requests in the vehicle schedule form a path in G, starting from s, ending at p and composed of T arcs. It proves that $\mathcal{I}_{EPP}$ is positive.

□

3.3 Intuitive description of the proof of NP-hardness of EPP

In order to prove that EPP is NP-Complete, 3-SAT is reduced to EPP. We first check that we may restrict ourselves to a 3-SAT instance (Z, C) such that for any variable $z_j, j = 1, \ldots, N$ the number $u(j)$ of occurrences of z_j in the clause collection $C = \{c_s, s = 1, \ldots, S\}$ is equal to the number of occurrences of $\neg z_j$ in C. It comes that if we set $U = \sum_j u(j)$, then we get $3S = 2N$. Then we build an acyclic Eulerian graph H^*, provided with a source s and a sink p, and do in such a way that paths from s to p which are likely to be involved in the related EPP instance will play the following role:

- For any clause $c_s = u_1 \vee u_2 \vee u_3$, where u_i denotes a literal, that means either some variable z_j or its negation $\neg z_j$, we shall have 3 significant paths which start from s and pass through node c_s, next traverse nodes related to variables and literals, and end by moving from some bottleneck node Q to node p: one of those paths, called **c-identifier** path, will assign the role of making c_s be equal to 1 to some literal u_i, while the 2 other paths, called **c-default** paths, will refuse this role to the 2 other literals. At the very end, $3S$ significant paths will be related this way to clause collection C.
- For any occurrence $1, \ldots, u(j)$ of variable z_j as a literal u_i inside some clause, we shall have 2 paths which start from s and go through node $(z_j, i), i = 1, \ldots, u(j)$, next traverse nodes related to variables and literals, and end by moving from copy node $(z_j, i)^*$ to node p: one of these paths, called **(j,u)-identifier** path, will tell us which value 0 or 1 takes z_i inside clause c_s, while the other one, called **(j,u)-default** path, will tell us which value is discarded. At the very end, $2N = 3S$ paths will be related this way the variables z_j.

Deriving such a path collection from a feasible solution Z of 3-SAT will come in a straightforward way. What will provide us with the converse, that means with the fact that any *ad hoc* partition of the arcs of H^* into admissible paths, will be a 2-sided device:

- The choice of the integral lengths $L(e)$ of the arcs e of graph H^*: since $6S$ paths must be built, the length of any path inside an EPP partition must be equal to $L = \sum_e \frac{L(e)}{6S}$. We are going to do in such a way that any path from s to p with length L is significant in the above sense, that means may be considered as **c-identifier**, **c-default**, **(j,u)-identifier** or **(j,u)-default**;
- A specific pattern, which we call saw pattern, which will ensure us that if some variable z_i takes the value 1 (or 0) inside some clause, then it takes the same value in every clause inside which it is involved.

As a matter of fact, the choice of length values $L(e)$ will be somewhat arbitrary, in the sense that it would be possible to choose other systems of length values, provided that above described significant paths keep being provided with length value $L = \sum_e \frac{L(e)}{6S}$ and that the lengths of non significant paths remain different from L.

Chapter 4

The Cumulative Cost Pickup and Delivery Problem on Rings

This chapter is based on the technical report:

- Trotta, M., Archetti, C., Feillet, D., and Quilliot, A. (2021). The cumulative cost pickup and delivery problem on rings. Working Paper MSE CMP-SFL 2021/10.

This chapter studies Pickup and Delivery Problems on Rings (PDP-R). PDP-R are defined on a circular network. A set of transportation requests has to be served. A fleet of vehicles is available at the depot. The objective is to assign requests to vehicles and define the service sequence for each vehicle, while minimizing the total completion time of requests. The problem is proved NP-hard. An ILP formulation is proposed and exhaustive computational tests are performed to show its effectiveness, comparing it with a straightforward greedy algorithm. Furthermore, a relax-and-repair heuristic based on a surrogate relaxation of the ILP formulation is proposed and compared with the greedy algorithm.

4.1 Introduction

Pickup and Delivery Problems on Rings (PDP-R) are defined on circular networks with m stations. A set of requests needs to be served, where each request consists of transporting a given quantity from a pickup station to a delivery station. A fleet of vehicles is available to perform the service and each vehicle route has to start and end at station 0, the depot. The goal is to determine the best transportation plan according to a given objective function.

PDP-R have been introduced in Trotta et al., 2022, where the authors propose a classification of the problem variants based on three fields $\alpha|\beta|\gamma$ where:

- α contains information about the number of vehicles and vehicle capacity;
- β specifies whether vehicles travel in one or both directions, whether release and due dates are defined and whether request have unitary demands;
- γ corresponds to the objective function.

In Trotta et al., 2022 applications of PDP-R are discussed: they are mainly linked to the use of autonomous vehicles for transportation services in concentrated areas (campus, industrial site, hospital...), for both people and freights. In addition, PDP-R also arise in the field of industrial automation. The reader is referred to Trotta et al., 2022 for a detailed overview of problem applications and related literature. The mostly related contributions are now briefly summarized, i.e., Pickup and Delivery Problems on circular graphs. As noted in Trotta et al., 2022, the literature is limited. In Guan, 1998 the multiple capacity nonpreemptive vehicle routing problem on cycles is studied. Gendreau et al., 1999 and Tzoreff et al., 2002 study freight transportation problems. Ilani et al., 2015, Pimenta et al., 2017a and Baïou et al., 2018 consider problems involving transportation of people, known as Dial-a-Ride problems.

In Trotta et al., 2022, the authors introduce the PDP-R, propose a classification scheme and study the variant in which vehicles travel on a single direction and the objective function is the minimization of the time at which the last vehicle returns to the depot. They also show that the simplest problem in this class, i.e., the problem with a single vehicle of unitary capacity and with no release and due date, is polynomially solvable while all others are NP-hard. In addition, they present a mathematical formulation for all variants which is proved to be effective through exhaustive computational tests.

Contrary to what is done in Trotta et al., 2022, this chapter focuses on the variants in which the objective function is the minimization of the sum of completion times of all requests. This objective is particularly relevant when people are transported because it reflects their average waiting time, and, so, the quality of the transportation service. It is showed that, in this case, even the simplest problem is NP-hard. In addition, a mathematical formulation for all problem variants and a greedy algorithm are proposed. Extensive computational tests show that the problem formulation is extremely effective in solving all problems variants, thus reducing the need for a fast sub-optimal (heuristic) approach. Also, the comparison with the greedy algorithm, which mimics common practice, shows the flaws of a non-optimized approach.

The chapter is organized as follows. Section 4.2, introduces the notation and some basic definitions. In Section 4.3 all the problems addressed in this chapter are proven NP-hard. A mathematical formulation is given in Section 4.5. Computational

experiments and the greedy algorithm are presented in Section 4.6. The results are summarized in Section 4.9, and some perspectives of this work are presented.

4.2 Problem description

PDP-R have been introduced in Trotta et al., 2022 (see also Chapter 1) and can be described as follows. The context is that of a circular layout (a ring) with m stations, numbered from 0 to $m-1$ in a clockwise direction. The distance from station j to station $j+1$ is denoted as $\delta_{j,j+1}$ and $L = \sum_{j=0}^{m-2} \delta_{j,j+1} + \delta_{m-1,0}$ is the total length of the ring. The problem consists in scheduling a set $\mathcal{R}$ of n transportation requests where each request $i \in \mathcal{R}$ asks for the transportation of a quantity of goods (or number of people) q_i from a pickup station s_i to a delivery station t_i. Each request specifies also a release date r_i and a due date d_i, that correspond, respectively, to the earliest time for pickup and the latest time for delivery. A fleet $\mathcal{V}$ of V vehicles of capacity Q is available at the depot (station 0) to serve the transportation requests. All the vehicles start their tours from the depot and get back to it when all requests are served. Vehicles are not allowed to wait at any station on the ring.

This chapter is focused on the study of PDP-R variants where the vehicles are allowed to move on the ring in a single direction (sd), i.e, either clockwise or counter-clockwise, and the objective function is the minimization of the total completion time ($\sum C_i$), where the completion time C_i is the time at which freight or people from request i are delivered at station t_i. According to the classification scheme introduced in Trotta et al., 2022, it is denoted $V, Q|sd, r_i, d_i|\sum C_i$. It is shown in the next section that even the variant with a single vehicle of capacity one and no release dates nor due dates is NP-hard $(1,1|sd,u|\sum C_i$ according to the classification, where u indicates that requests are unitary). It follows, that the more general problem is also NP-hard.

In the following, the terms *ring* and *circle* will be used indifferently to represent the underlying network structure. Without loss of generality, we assume that all vehicles go clockwise. We say that two requests $i_1, i_2 \in \mathcal{R}$ *overlap* if at least one segment $[j, j+1]$ of the ring is covered by both requests ($j = 0, ..., m-1$). We denote by $\delta_{j_1 j_2}$ the traveled distance between two stations j_1 and j_2:

$$\delta_{j_1,j_2} = \begin{cases} \sum_{j=j_1}^{j_2-1} \delta_{j,j+1} & \text{if } j_1 < j_2 \\ \sum_{j=j_1}^{m-1} \delta_{j,j+1} + \sum_{j=0}^{j_2-1} \delta_{j,j+1} & \text{otherwise} \end{cases}$$

In this single-direction context, it is interesting to see that completion time C_i can be matched to the number of ring tours completed before starting serving request $i \in \mathcal{R}$ at station s_i:

$$C_i = L \times l_i + \delta_{0,s_i} + \delta_{s_i,t_i} \tag{4.1}$$

where l_i is a decision variable indicating this number of tours.

4.3 Complexity of problem $1,1|sd,u|\sum C_i$

In this section, problem $1,1|sd,u|\sum C_i$ is proven NP-hard. Note that all remaining problem variants considered in this chapter are generalizations of $1,1|sd,u|\sum C_i$ and are thus also NP-hard.

Before proving the computational complexity of the problem, note that minimizing the sum of completion times corresponds to minimizing $\sum_{i\in\mathcal{R}} l_i$, as can be seen in (4.1). The following proof is based on the minimization of the latter.

We proceed by reduction from the Sum Coloring Problem (SCP) that is defined as follows:

Definition 11. *Given a graph $G = (V, E)$ with n nodes, find a coloring of the nodes such that the sum of the colors assigned to nodes is minimum.*

The decision version of SCP consists in determining whether there exists a coloring such that the sum of the colors assigned to the nodes is smaller than or equal to a given positive integer K. The minimum value for the sum of the colors is known as the Chromatic Sum. In Szkaliczki, 1999 and Marx, 2005, the problem is shown to be NP-complete also in the case where the underlying graph is an interval graph. In the following, we see a reduction from the decision version of the SCP on an interval graph to $1,1|sd,u|\sum C_i$.

Theorem 8. *Problem $1,1|sd,u|\sum C_i$ is NP-hard.*

Proof. The main argument in the proof is that when no request covers the depot, the circle is equivalent to an interval graph, where intervals represent requests. The tour in which a request is served can be interpreted as a color, and so minimizing the sum of the tour numbers is equivalent to minimizing the sum of the colors. Given that fact, we first show how to transform an instance of the decision version of SCP on an interval graph into an instance of $1,1|sd,u|\sum C_i$.

We consider an instance I_c of the decision version of SCP on an interval graph. $\mathcal{A} = \{(a_1, b_1), ..., (a_n, b_n)\}$ is a set of n intervals on a real line with $a_i < b_i$ for $i = 1, \ldots, n$. An interval graph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ is constructed from $\mathcal{A}$ by introducing a vertex v_i for each interval $(a_i, b_i) \in \mathcal{A}$ and by adding an edge (v_i, v_j) each time intervals (a_i, b_i) and (a_j, b_j) overlap. Instance I_c consists in deciding if, given an integer K, there exists a coloring function $f : \mathcal{V} \to \mathbb{N}$ such that $\sum_{v\in\mathcal{V}} f(v) \leq K$.

We build an instance I_s of $1,1|sd,u|\sum C_i$ as follows. Let $\mathcal{D}$ denote the set of interval extremities of $\mathcal{A}$: $\mathcal{D} = \{a_i, b_i : i = 1, ..., n\}$. We sort $\mathcal{D}$ in increasing order

and introduce a station in the ring for every element in $\mathcal{D}$, in this order. We complete the ring with station 0, the depot. This way, the number m of stations is at most two times the number n of intervals plus the depot ($m \leq 2n + 1$). We define unitary distances between successive stations, which gives a tour length $L = |\mathcal{D}| + 1$. We introduce a request in $\mathcal{R}$ for every interval in $\mathcal{A}$. Given interval (a_i, b_i), request i is defined as follows: s_i is the station obtained from extremity a_i, t_i is the station obtained from extremity b_i. We note that no request covers the depot, so two requests can be scheduled in the same tour if and only if they do not overlap. Note also that this construction is polynomial.

We claim that I_c is positive if and only if I_s admits a K-feasible schedule. A K-feasible schedule for I_s is a function $f : \mathcal{R} \to \mathbb{N}$ that assigns each request $i \in \mathcal{R}$ to a tour (that is the tour in which the request is served) so that requests do not overlap and that the sum of the tour numbers of all requests is not greater than K. More formally, function f is such that:

- $f(i_1) = f(i_2) \implies i_1$ and i_2 do not overlap $\forall i_1, i_2 \in \mathcal{R}$
- $\sum_{i \in \mathcal{R}} f(i) \leq K$.

Assume first that I_c is positive. This means that there exists a coloring function $g : \mathcal{V} \to \mathbb{N}$ for $\mathcal{H}$ such that $\sum_{v \in \mathcal{V}} g(v) \leq K$. Let $f : \mathcal{R} \to \mathbb{N}$ be a function such that $f(i) = g(v_i)$, $v_i \in \mathcal{V}$. It is easy to see that f is a K-feasible schedule for I_s.

On the other hand, assume that I_s admits a K-feasible schedule f. Let $g : \mathcal{V} \to \mathbb{R}$ be a function such that $g(v_i) = f(i)$, $(a_i, b_i) \in \mathcal{A}$. It is easy to see that g is a coloring function for $\mathcal{H}$ such that $\sum_{v_i \in \mathcal{V}} g(v_i) \leq K$. In fact, $g(v_{i_1}) = g(v_{i_2})$ means that requests i_1 and i_2 do not overlap, *i.e.,* edge $(v_{i_1}, v_{i_2}) \notin \mathcal{E}$. In addition, since $g(v_i) = f(i)$ for each $i \in \mathcal{R}$, $\sum_{v_i \in \mathcal{V}} g(v_i) = \sum_{i \in \mathcal{R}} f(i) \leq K$.

Thus, solving SCP on an interval graph amounts to finding a feasible solution to an instance of problem $1, 1|sd, u| \sum C_i$. This proves that problem $1, 1|sd, u| \sum C_i$ is NP-hard. □

4.4 A simple proof of NP-completeness of CACP

The aim of this section is twofold. A simple proof of NP-completeness of the Circular Arc Coloring Problem (CACP), that was introduced and proven to be NP-complete in Garey et al., 1980, is presented. Furthermore, a special case of CACP is proven NP-hard by reduction from the Arc-Disjoint Path Problem. This proves that the CACP is NP-complete even for this special case. The proof is similar to the one given in Marx, 2003 which was not known at the time this proof was written.

We first need some definitions in order to introduce the CACP.

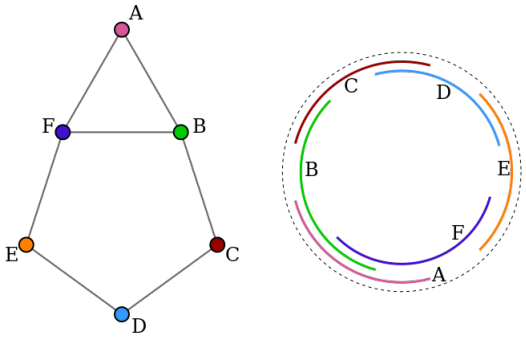


Figure 4.1: A circular-arc graph (left) and the corresponding arc model (right).

Definition 12. *A circular arc graph G is a graph whose vertices can be placed in one-to-one correspondence with a family F of arcs of a circle in such a way that two vertices of G are joined by an edge if and only if the corresponding two arcs in F intersect one another. It is therefore the intersection graph of intervals on a circle (see figure 4.1).*

In order for the arcs to intersect, the intersection of the points that they cover on the circle must not be empty nor reduced to 1 point (or 2 points, in case two arcs are one the complement of each other).

Definition 13. *A family F of circular arcs is a set $\{A_1, A_2, ..., A_n\}$, where each A_i is an ordered pair (a_i, b_i) of positive integers, with $a_i \neq b_i$.*

Let m denote the largest integer among all the a_i's and b_i's. Then we can regard the circle as being divided into m parts by m equally spaced points, numbered clockwise as $0, 1, \ldots, m-1$, and each $A_i = (a_i, b_i)$ can be regarded as representing the circular arc from point a_i to point b_i, again in the clockwise direction. Notice that we might have either $a_i < b_i$ or $b_i < a_i$ for any A_i. The circular arc graph corresponding to the family F is the graph $G = (F, E)$, where $\{A_i, A_j\} \in E$ if and only if A_i and A_j intersect.

What follows is the definition of the Circular Arc Coloring problem.

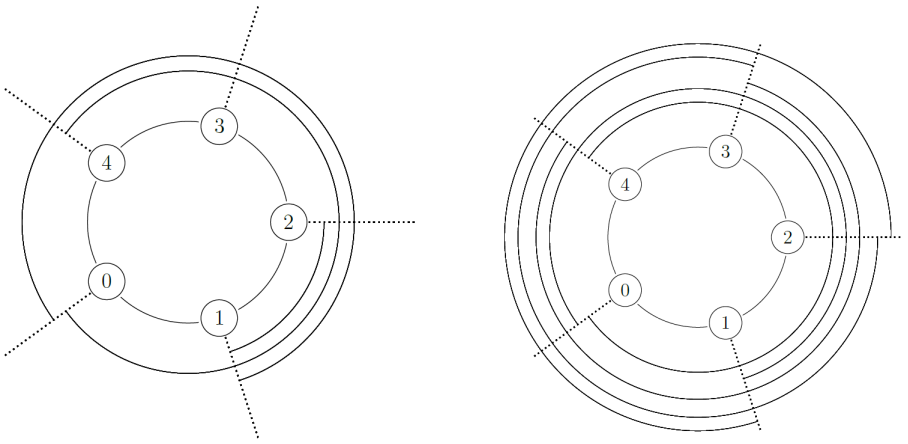
Definition 14. *Given a family F of circular arcs and a positive integer K, can F be partitioned into K classes so that no two arcs in the same class intersect? Or, equivalently, can the circular arc graph $G = (F, E)$ be colored with K colors?*

If yes, we say the family F or graph G are K-colorable. As mentioned at the beginning of this section, the CACP is NP-complete (Garey et al., 1980; Marx, 2003). It is now presented a simpler proof of NP-completeness of the CACP by proving the NP-hardness of a special case.

We first need some definitions in order to define the special case of the CACP.

Definition 15. *The Linear Hypothesis (LH) corresponds to the case where point* 0 *on the circle is contained into no open interval* $]a_i, b_i[$ *(see figure 4.2(a)).*

Definition 16. *The Fillness Hypothesis (FH) corresponds to the case where every segment* $[j, j+1]$ $(j = 0, ..., m-1)$ *of the circle is covered by a constant number of arcs. We call this constant the Fillness Coefficient (FC) see figure 4.2(b)).*



(a) Linear Hypothesis

(b) Fillness Hypothesis with $FC = 4$

Figure 4.2: Examples of circles with LH (on the left) and FH (on the right).

Property 1. *If LH holds, we might open the circle which means that we split point 0 into two distinct points 0 and* m*, and consider any interval* $[a_i, b_i]$ *as a sub-interval of the interval* $[0, m]$*. In other words, the circular arc graph is equivalent to the interval graph where point* 0 *is split in two points.*

Property 2. *If both LH and FH hold, then graph G may clearly be colored with FC colors (see Figure 4.2 (b)).*

Proof. If both LH and FH hold, graph G is an interval graph where the number of pairwise overlapping intervals is constant and equal to FC. The problem of coloring interval graphs is known as the Interval Graph Coloring Problem or Interval Partitioning Problem, and it can be optimally solved by applying a greedy algorithm that uses exactly FC colors (see, for example, Kleinberg and Tardos, 2006, pp. 122-125). □

We now consider the problem of coloring a circular-arc graph G where only FH holds, *i.e.* the problem of finding a coloring G with exactly FC colors (also known as FC-coloring). We call this problem the Full Circular-Arc Coloring Problem.

We want to prove that the NP-completeness of CACP also holds for FCACP, that is when *FH* is satisfied. To do so, the Arc Disjoint Path Problem (ADPP) is reduced to the the FCACP.

The ADPP is defined as follows:

Definition 17. *Let $G = (X, E)$, be a directed acyclic graph, given together with n origin vertices $s_1, ..., s_n \in X$, and n destination vertices $t_1, ..., t_n \in X$. The problem asks if there exist n paths $\Gamma_1, ..., \Gamma_n$, pairwise arc-disjoint, and such that, for any $j = 1, ..., n$, Γ_j starts from s_j and arrives to t_j.*

This problem is known as the *Arc Disjoint Path* problem or the *weak k-LINKAGE* problem, and ah showed in Even et al., 1976 it is NP-complete even for directed acyclic graphs if the number of paths is given as input.

FCACP can be now proven NP-Complete. The NP membership comes from the fact that FCACP is a special case of CACP, so we just need to prove the NP-hardness by reducing ADPP to FCACP.

We start from an instance $\mathcal{I} = (G = (X, E), s_1, ..., s_n, t_1, .., t_n)$ of the *Arc Disjoint Path Problem* (see figure 4.3).

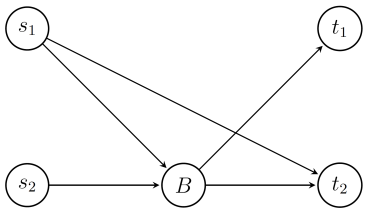


Figure 4.3: A positive instance of the Arc Disjoint path problem with $n = 2$

Then we sort the nodes in X by calculating a topological sorting (we can do it because G is a DAG) and consider the nodes in this order. We associate each node with an integer label $H(x)$ from 1 to $|X|$ on the real line by following the topological sorting.

We then consider the interval $[0, m]$ on the real line, where $m = |X| + 1$, and we roll up the interval by transforming it into a circle in such a way that one can ideally walk clockwise on the circle to go from 0 to m (see figure 4.4). We merge points 0 and m into one reference point 0 and we get an oriented circle Γ. We define on Γ an interval collection I as follows:

- For any arc (x, y) of graph G, we set $I_{x,y}$ = the interval $[H(x), H(y)]$ which we get by going from $H(x)$ to $H(y)$ while following the orientation of Γ;

- For any origin-destination pair (s_j, t_j), we set I_j^* = the interval $[H(t_j), H(s_j)]$ which we get by going from $H(t_j)$ to $H(s_j)$ while following the orientation of Γ.

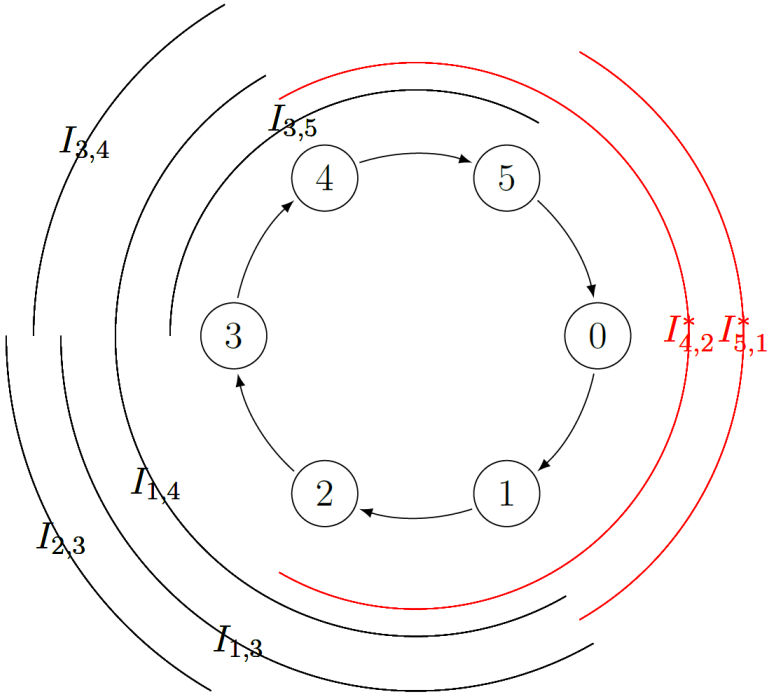


Figure 4.4: Oriented circle Γ and intervals corresponding to the instance of ADPP in figure 4.3. Legend: $s_1 = 1, s_2 = 2, B = 3, t_2 = 4, t_1 = 5$. Intervals corresponding to the requested paths are depicted in red, while the graph arcs are depicted in black.

Observation 1. *For any origin-destination pair (s_j, t_j) every interval I_j^* contains point 0.*

Property 3. *If we denote with Γ_j the path going from s_j to t_j in graph G and with I_{Γ_j} the collection of consecutive intervals on the circle Γ corresponding to the arcs of path Γ_j, then the collection of intervals $I_{\Gamma_j} \cup \{I_j^*\}$ completely covers the circle.*

Proof. Collection I_{Γ_j} contains by construction all and only the points on the circle that are between s_j and t_j in clockwise order. Collection I_j^* contains all and only the points on the circle that are between t_j and s_j in clockwise order. Their union must therefore cover all the circle. □

For any $x \in X$, we denote by $d^+(x)$ the number of intervals of the collection $I = \{I_{x,y}, (x,y) \text{ arcs of } G\} \cup \{I_j^*, j = 1, ..., n\}$ which start from x. We denote by

$d^-(x)$ the number of intervals of the collection $\{I_{x,y}, (x,y)$ arcs of $G\} \cup \{I_j^*, j = 1, ..., n\}$ which end into x. If $d^+(x) < d^-(x)$, then we insert $d^-(x) - d^+(x)$ copies L_x^k $(k = 1, ..., d^-(x) - d^+(x))$ of the interval $[H(x), 0]$ into I. If $d^+(x) > d^-(x)$, then we insert $d^+(x) - d^-(x)$ copies L_x^{*k} $(k = 1, ..., d^+(x) - d^-(x))$ of the interval $[0, H(x)]$ into I. At the end we have that for every integer node x, $d^+(x) = d^-(x)$ (see figure 4.5).

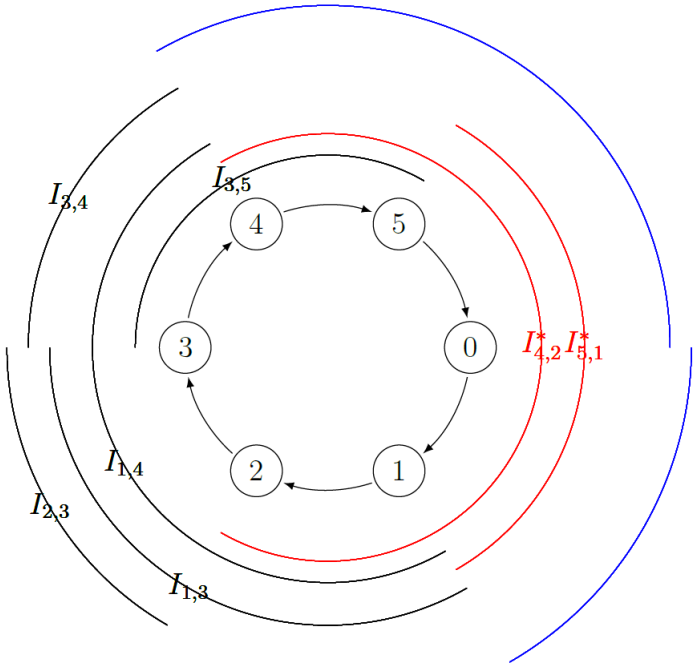


Figure 4.5: The augmented collection of intervals. The added intervals are depicted in blue.

Lemma 7 proves that the reduction is correct (*i.e.* that I is an instance of FCACP) and lemma 8 proves that ADPP reduces to FCACP.

Lemma 7. *The augmented collection* $I = \{I_{x,y}, (x,y) \in E\} \cup \{I_j^*, j = 1..n\} \cup \{L_x^k,\ x$ *such that* $d^+(x) < d^-(x),\ k = 1, ..., d^-(x) - d^+(x)\} \cup \{L_x^{*k},\ x$ *such that* $d^+(x) > d^-(x),\ k = 1, ..., d^-(x) - d^+(x)\}$ *satisfies FH; we denote by* K *the related Fillness Coefficient).*

Proof. Let us consider two consecutive integer nodes x and $x+1$ on the circle. The number of intervals that cover the segment $[x, x+1]$ is equal to the number of intervals that start from x plus $C(x) =$ the number of intervals that cover x. This number is also equal to the number of intervals that end into $x+1$ plus $C(x+1) =$ the number of intervals that cover $x+1$. So we have that $d^+(x) + C(x) = d^-(x+1) + C(x+1)$. Since $d^+(x) = d^-(x)\ \forall x \in x$, we have that $d^-(x) + C(x) = d^+(x) + C(x) = d^+(x +$

$1) + C(x+1)$. Since we made no hypotheses on x and $x+1$, this holds for any x. This means that the number of intervals between any pair of nodes is the same, and we denote by K this quantity. □

Lemma 8. *The existence of n pairwise arc disjoint paths $\Gamma_1, ..., \Gamma_n$ in graph G, such that for any j $(j = 1, ..., n)$ Γ_j starts from s_j and arrives to t_j, is equivalent to the existence of a K-coloring of the collection of intervals I.*

Proof. Let $\Gamma_1, ..., \Gamma_n$ be n pairwise arc disjoint paths in graph G such that for any j $(j = 1, ..., n)$, Γ_j starts from s_j and arrives to t_j. These paths may be viewed as disjoint collections of intervals of I, each Γ_j corresponding to a set of consecutive intervals $I_{x_1,x_2}, I_{x_2,x_3}..., I_{x_{r-1},x_r}$ where $x_1 = s_j$, $x_r = t_j$ and two consecutive intervals being $I_{x_i,x_{i+1}}, I_{x_{i+1},x_{i+2}}$ for any $i = 1, ..., r-2$. We denote by I_{Γ_j} the subset of intervals of I on circle Γ that correspond to path Γ_j in graph G. To each augmented collection $I_{\Gamma_j} \cup \{I_j^*\}, (j = 1, \ldots, n)$ we can assign color j because all intervals in this collection do not overlap on the circle (for property 3). Remaining intervals are sub-intervals of the interval $[0, m]$ (point 0 is not contained in any of the remaining intervals because of the way we constructed the collection) and therefore satisfy LH. This collection satisfies also FH for lemma 7, with $FC = K - n$ (because n colors have been already used) and for property 2 admits a $(K-n)$-coloring. Thus I admits a K-coloring.

Conversely, let τ be a K-coloring of I. For property 3, we may suppose without loss of generality that every collection $I_{\Gamma_j} \cup \{I_j^*\}$ has color j, $(j = 1, \ldots, n)$ as they completely cover circle Γ. Therefore each interval I_j^* has a different color $\tau(I_j^*) = j$, $j = 1, \ldots, n$. Let us consider, for any $j = 1, \ldots, n$, intervals related to color j: they are of the form $I_{x_1,x_2}, I_{x_2,x_3}..., I_{x_{r-1},x_r}$ where $x_1 = s_j$, $x_r = t_j$ and two consecutive intervals being $I_{x_i,x_{i+1}}, I_{x_{i+1},x_{i+2}}$ for any $i = 1, ..., r-2$. Each of them corresponds to a path Γ_j from s_j to t_j $(j = 1, ..., n)$ in graph G and all these paths are arc-disjoint because all intervals I_j^* have different colors. □

Theorem 9. *FCACP is NP-complete.*

Proof. The theorem follows from lemmas 7 and 8 and from the fact that the reduction is polynomial. □

4.5 An ILP formulation

In this section it is presented an ILP formulation for the most general variant of the classes of problems studied in this chapter, i.e., $V, Q|sd, r_i, d_i|\sum C_i$. As already observed, the objective can be expressed as the minimization of the sum of the number of vehicle tours. The formulation makes use of the following notation, in addition to the one introduced in previous sections.

- $\mathcal{R}[j, j+1]$: set of requests in $\mathcal{R}$ that cover segment $[j, j+1]$. A request $i \in \mathcal{R}$ covers segment $[j, j+1]$ if stations j and $j+1$ are within stations s_i and t_i when going clockwise. Equivalently, it means that a vehicle needs to traverse arc $(j, j+1)$ in order to serve i.
- c_{ik}: cost of inserting request $i \in \mathcal{R}$ in tour k

$$c_{ik} = \begin{cases} k & \text{if } s_i < t_i \\ k+1 & \text{otherwise} \end{cases}$$

- $\mathcal{K}_i$: set of tour numbers in which request i can be started

$$\mathcal{K}_i = \{k \in \mathbb{N} : (k-1)L + \delta_{0,s_i} \geq r_i \text{ and } (k-1)L + \delta_{0,s_i} + \delta_{s_i,t_i} \leq d_i\}.$$

- K: upper bound on the tour number for the request that will be started last

$$K = \max_{1 \leq i \leq n} \left(\left\lceil \frac{d_i}{L} \right\rceil \right).$$

We introduce the following decision variables:

$$x_{ikv} = \begin{cases} 1 & \text{if request } i \text{ is started in tour } k \text{ by vehicle } v \\ 0 & \text{otherwise.} \end{cases}$$

The integer linear program is then:

$$\min \sum_{i \in \mathcal{R}} \sum_{v \in \mathcal{V}} \sum_{k \in \mathcal{K}_i} c_{ik} x_{ikv} \tag{4.2}$$

s.t.:

$$\sum_{\{i \in \mathcal{R}[j,j+1], s_i \leq j\}} q_i x_{ikv} + \sum_{\{i \in \mathcal{R}[j,j+1], s_i > j+1\}} q_i x_{ik-1v} \leq Q \tag{4.3}$$

$$\sum_{v \in \mathcal{V}} \sum_{k \in \mathcal{K}_i} x_{ikv} = 1 \tag{4.4}$$

$$x_{ikv} \in \{0, 1\} \tag{4.5}$$

with $1 \leq i \leq n, 0 \leq j \leq m-1, 1 \leq k \leq K, 1 \leq v \leq V$.

The objective function minimizes the sum of completion times. Constraints (4.3) ensure that vehicle capacities are satisfied: for each vehicle, each tour and each interval $[j, j+1]$, the quantity loaded in the vehicle is not greater than Q. This quantity is obtained by considering requests that cover $[j, j+1]$ and by summing

quantities q_i of requests that started being served in this tour before station j and requests that started being served in the previous tour after station $j+1$. Constraints (4.4) make sure that every requests is served, at an acceptable time. Constraints (4.5) define variable domains.

Two types of symmetry breaking constraints were tested in order to strengthen the formulation. The first ranks vehicles according to the number of satisfied requests: the smallest the vehicle index, the largest the number of satisfied requests

$$\sum_{i\in\mathcal{R}}\sum_{k\in\mathcal{K}_i} x_{ikv} \geq \sum_{i\in\mathcal{R}}\sum_{k\in\mathcal{K}_i} x_{ikv+1} \qquad (1 \leq v \leq V-1)$$

The second set of symmetry breaking constraints imposes that first requests are served by first vehicles: the i first requests have to be assigned to vehicles in set $\{1,\ldots,i\}$:

$$\sum_{u=1}^{i}\sum_{v=1}^{i}\sum_{k\in\mathcal{K}_u} x_{ukv} = i \qquad (1 \leq i \leq \min(V,n))$$

Note that the two constraint sets cannot be considered simultaneously. Their impact was evaluated computationally and a slight improvement of computational times was noticed on some instances. Anyway, on average, the model performs better without the symmetry-breaking constraints, so they are not included in the following computational tests.

4.6 Computational tests

It is now presented the set of experiments made in order to evaluate the efficacy of formulation (4.2)–(4.5). The formulation is solved through CPLEX 12.9.0 on a computer equipped with an Intel Core i7-9700 processor and 32GB of RAM. For all experiments, a time limit of 30 minutes was set. All instances were solved within this time limit.

To the best of our knowledge, the problems investigated in this chapter are new. Thus, the computational tests have been made on instances proposed in Trotta et al., 2022 for PDP-R with a different objective function. These instances are defined on a circle with $m = 10$ stations, $n = 20$, 40, 80 or 160 requests, $V = 1$, 2 or 3 vehicles and $Q = 1$, 2, 4 or 8. Pickup and delivery stations are randomly drawn, as well as demands (in $\{1,\ldots,Q\}$ for the latter). When needed, release dates and due dates are defined according to four categories: t-t, t-w, w-t, w-w, where t and w stand for tight and wide, respectively, and the two letters concern release dates and due dates, respectively. Five instances are generated for each combinations of parameters. The reader is referred to Trotta et al., 2022 for more details concerning these instances.

4.6.1 A greedy algorithm

As a comparison basis with respect to the exact solution with a solver, it was designed a greedy algorithm that is easy to implement and that sounds natural for practitioners. It works as follows.

When a vehicle reaches a station that is the pickup station of at least one request and this request can be started, it is started. If several requests can be started at the same time, ties are broken by giving priority to requests according to the following criteria, taken hierarchically:

- earliest due date first (due date is considered infinite if no due date is defined)
- shortest distance to reach the delivery node first
- largest demand first

Note that, in case of instances with unit demands and no release/due dates, only the second criterion is used.

4.6.2 Computational results for $V, Q|sd, u|\sum C_i$

Table 4.1 reports results for problem $V, Q|sd, u|\sum C_i$, i.e., with no release and due dates and unitary demands. All results are averaged over the 5 instances with the same value of n, V and Q. Column *CPU* reports the solution time of the formulation in Section 4.5, in seconds. Column *feas* reports the number of instances for which a feasible solution has been found. In all our experiments, infeasibility was proven each time no feasible solution was found. For this reason this information is not reported in the tables: instances for which no feasible solution has been found are infeasible. Column *opt* reports the number of instances solved to optimality, among those that admit a feasible solution. Column *gr* reports the number of instances for which the greedy algorithm found a feasible solution. Column $\sum C_i$ reports, with a slight abuse of notation, the value of the best solution found within the time limit. Column gap^O reports the percentage relative gap between the solution of the greedy algorithm and the one of the formulation, related to instances solved to optimality, while column gap^{UB} reports the gap related to instances for which the formulation found a feasible (but not necessarily optimal) solution. Precisely, gap^O and gap^{UB} are calculated as $\frac{x_{gr}-x_{opt}}{x_{opt}}$ and $\frac{x_{gr}-x_{feas}}{x_{feas}}$ respectively, where x_{gr}, x_{opt} and x_{feas} are the greedy, optimal and feasible solution values.

Formulation (4.2)–(4.5) is extremely effective: all instances are solved to optimality, with the largest computing time being a few seconds. The greedy algorithm is also capable of solving all instances. The gaps it provides with respect to optimal

n	V	Q	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$
20	1	1	0.0	5	5	5	114.4	6.3	-
20	1	2	0.0	5	5	5	66.6	5.4	-
20	1	4	0.0	5	5	5	43.4	4.2	-
20	1	8	0.0	5	5	5	33.2	1.4	-
20	2	1	0.0	5	5	5	66.6	5.4	-
20	2	2	0.0	5	5	5	43.4	4.2	-
20	2	4	0.0	5	5	5	33.2	1.4	-
20	2	8	0.0	5	5	5	28.6	0.0	-
20	3	1	0.0	5	5	5	51.0	4.7	-
20	3	2	0.0	5	5	5	36.0	1.3	-
20	3	4	0.0	5	5	5	29.4	0.0	-
20	3	8	0.0	5	5	5	28.6	0.0	-
40	1	1	0.1	5	5	5	404.0	5.6	-
40	1	2	0.0	5	5	5	216.2	5.4	-
40	1	4	0.0	5	5	5	124.6	4.0	-
40	1	8	0.0	5	5	5	81.0	1.4	-
40	2	1	0.1	5	5	5	216.2	5.4	-
40	2	2	0.0	5	5	5	124.6	4.0	-
40	2	4	0.0	5	5	5	81.0	1.4	-
40	2	8	0.0	5	5	5	61.6	0.0	-
40	3	1	0.1	5	5	5	154.8	5.2	-
40	3	2	0.0	5	5	5	94.6	3.1	-
40	3	4	0.0	5	5	5	67.4	0.3	-
40	3	8	0.0	5	5	5	56.2	0.0	-
80	1	1	0.4	5	5	5	1404.2	4.8	-
80	1	2	0.2	5	5	5	733.2	4.8	-
80	1	4	0.1	5	5	5	398.8	5.0	-
80	1	8	0.1	5	5	5	234.8	4.7	-
80	2	1	0.4	5	5	5	733.2	4.8	-
80	2	2	0.2	5	5	5	398.8	5.0	-
80	2	4	0.1	5	5	5	234.8	4.7	-
80	2	8	0.1	5	5	5	155.8	4.1	-
80	3	1	0.4	5	5	5	510.0	4.9	-
80	3	2	0.2	5	5	5	289.0	5.4	-
80	3	4	0.1	5	5	5	181.4	4.1	-
80	3	8	0.1	5	5	5	130.2	2.6	-
160	1	1	3.0	5	5	5	5014.2	1.4	-
160	1	2	1.3	5	5	5	2565.0	1.4	-
160	1	4	0.9	5	5	5	1343.2	1.8	-
160	1	8	0.3	5	5	5	737.4	2.4	-
160	2	1	3.3	5	5	5	2565.0	1.4	-
160	2	2	1.1	5	5	5	1343.2	1.8	-
160	2	4	0.9	5	5	5	737.4	2.4	-
160	2	8	0.4	5	5	5	441.2	2.7	-
160	3	1	2.4	5	5	5	1749.6	1.4	-
160	3	2	1.2	5	5	5	938.6	1.9	-
160	3	4	1.0	5	5	5	539.8	2.7	-
160	3	8	0.3	5	5	5	346.4	1.4	-

Table 4.1: Problem $V, Q|sd, u|\sum C_i$

solutions only go up to slightly more than 6%. Furthermore, these gaps are even smaller for large vehicle capacities: this might be due to the fact that capacity is less binding in this case and, thus, the choice of favoring requests with shorter distance to destination pays off.

4.6.3 Computational results for $V, Q|sd, u, r_i, d_i|\sum C_i$

Tables 4.2 and 4.3 report results for problem $V, Q|sd, u, r_i, d_i|\sum C_i$ i.e., the case with release and due dates and unitary demands. As before, each row reports the average value over the 5 instances with the same value of n, V and Q. In Table 4.2, three cases are considered for release and due dates: tight-tight (t-t), tight-wide (t-w) and tight release dates without due dates (t). The three other cases are reported in Table 4.3.

n	V	Q	t-t							t-w							t						
			CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$
20	1	1	0.0	3	3	0	198.3	-	-	0.0	4	4	1	193.3	9.2	-	0.0	5	5	5	196.8	8.1	-
20	1	2	0.0	5	5	0	108.0	-	-	0.0	5	5	0	106.8	-	-	0.0	5	5	5	107.8	13.5	-
20	1	4	0.0	5	5	0	57.2	-	-	0.0	5	5	1	58.4	25.4	-	0.0	5	5	5	57.2	19.9	-
20	1	8	0.0	5	5	0	36.4	-	-	0.0	5	5	2	39.4	38.1	-	0.0	5	5	5	36.4	37.9	-
20	2	1	0.0	5	5	0	97.6	-	-	0.0	5	5	0	100.0	-	-	0.0	5	5	5	97.4	15.2	-
20	2	2	0.0	5	5	2	57.8	20.1	-	0.0	5	5	2	55.0	29.6	-	0.0	5	5	5	57.8	20.9	-
20	2	4	0.0	5	5	0	39.0	-	-	0.0	5	5	0	39.4	-	-	0.0	5	5	5	39.0	31.2	-
20	2	8	0.0	5	5	0	28.6	-	-	0.0	5	5	0	28.6	-	-	0.0	5	5	5	28.6	40.7	-
20	3	1	0.0	5	5	2	71.8	19.4	-	0.0	4	4	2	73.0	16.9	-	0.0	5	5	5	71.6	18.6	-
20	3	2	0.0	5	5	0	46.0	-	-	0.0	5	5	2	44.8	10.2	-	0.0	5	5	5	46.0	21.8	-
20	3	4	0.0	5	5	0	32.4	-	-	0.0	5	5	1	33.0	46.7	-	0.0	5	5	5	32.4	30.7	-
20	3	8	0.0	5	5	0	28.6	-	-	0.0	5	5	0	28.6	-	-	0.0	5	5	5	28.6	31.7	-
40	1	1	0.0	5	5	0	738.8	-	-	0.0	5	5	2	635.2	7.7	-	0.0	5	5	5	736.0	4.9	-
40	1	2	0.0	5	5	0	367.4	-	-	0.0	5	5	0	362.0	-	-	0.0	5	5	5	366.2	7.3	-
40	1	4	0.0	5	5	0	191.6	-	-	0.0	5	5	1	181.2	8.7	-	0.0	5	5	5	191.6	12.7	-
40	1	8	0.0	5	5	0	112.6	-	-	0.0	5	5	0	116.2	-	-	0.0	5	5	5	112.6	16.3	-
40	2	1	0.0	5	5	0	353.4	-	-	0.0	5	5	0	339.4	-	-	0.1	5	5	5	351.0	9.1	-
40	2	2	0.0	5	5	1	197.6	10.1	-	0.0	5	5	3	190.6	12.7	-	0.0	5	5	5	197.0	11.8	-
40	2	4	0.0	5	5	0	109.0	-	-	0.0	5	5	0	112.2	-	-	0.0	5	5	5	109.0	19.6	-
40	2	8	0.0	5	5	0	74.0	-	-	0.0	5	5	0	73.4	-	-	0.0	5	5	5	74.0	30.2	-
40	3	1	0.0	5	5	0	242.4	-	-	0.0	5	5	1	243.8	10.0	-	0.1	5	5	5	242.0	11.3	-
40	3	2	0.0	5	5	0	144.8	-	-	0.0	5	5	0	145.8	-	-	0.0	5	5	5	144.8	15.9	-
40	3	4	0.0	5	5	0	84.4	-	-	0.0	5	5	0	88.0	-	-	0.0	5	5	5	84.4	22.0	-
40	3	8	0.0	5	5	0	65.4	-	-	0.0	5	5	0	63.4	-	-	0.0	5	5	5	65.4	29.4	-
80	1	1	0.1	5	5	1	2490.6	3.2	-	0.1	4	4	0	2495.0	-	-	0.1	5	5	5	2489.6	3.6	-
80	1	2	0.0	5	5	0	1303.4	-	-	0.0	5	5	0	1304.8	-	-	0.1	5	5	5	1302.8	4.9	-
80	1	4	0.0	5	5	1	700.2	7.9	-	0.0	5	5	1	673.8	8.6	-	0.1	5	5	5	699.8	7.2	-
80	1	8	0.0	5	5	0	367.6	-	-	0.0	5	5	1	372.2	14.9	-	0.1	5	5	5	367.6	13.6	-
80	2	1	0.1	5	5	1	1260.0	4.3	-	0.1	5	5	0	1319.4	-	-	0.2	5	5	5	1259.4	5.8	-
80	2	2	0.0	5	5	0	672.6	-	-	0.0	5	5	1	678.6	9.9	-	0.1	5	5	5	672.4	8.5	-
80	2	4	0.0	5	5	2	382.2	13.7	-	0.0	5	5	0	361.8	-	-	0.1	5	5	5	382.2	12.9	-
80	2	8	0.0	5	5	0	228.0	-	-	0.0	5	5	0	222.6	-	-	0.1	5	5	5	228.0	18.5	-
80	3	1	0.1	5	5	1	896.0	7.3	-	0.1	5	5	1	907.6	4.9	-	0.3	5	5	5	895.4	6.9	-
80	3	2	0.0	5	5	1	462.0	9.7	-	0.0	5	5	1	485.6	9.3	-	0.2	5	5	5	462.0	10.3	-
80	3	4	0.0	5	5	0	270.8	-	-	0.0	5	5	0	266.0	-	-	0.2	5	5	5	270.8	18.4	-
80	3	8	0.0	5	5	0	181.0	-	-	0.0	5	5	0	177.6	-	-	0.1	5	5	5	181.0	23.5	-
160	1	1	0.3	5	5	0	9889.4	-	-	0.3	4	4	0	10137.8	-	-	0.6	5	5	5	9886.0	1.9	-
160	1	2	0.1	5	5	1	4922.8	2.6	-	0.1	5	5	1	5122.0	2.0	-	0.3	5	5	5	4921.2	2.4	-
160	1	4	0.0	5	5	1	2521.4	4.3	-	0.1	5	5	0	2622.0	-	-	0.2	5	5	5	2521.4	3.9	-
160	1	8	0.0	5	5	0	1389.0	-	-	0.0	5	5	2	1356.8	5.9	-	0.2	5	5	5	1388.8	6.5	-
160	2	1	0.2	5	5	0	5042.8	-	-	0.4	5	5	0	5015.4	-	-	0.9	5	5	5	5042.8	2.4	-
160	2	2	0.1	5	5	0	2564.2	-	-	0.1	5	5	1	2551.8	3.3	-	0.6	5	5	5	2563.6	3.6	-
160	2	4	0.0	5	5	1	1359.0	7.5	-	0.0	5	5	0	1323.0	-	-	0.5	5	5	5	1359.0	6.8	-
160	2	8	0.0	5	5	0	743.0	-	-	0.0	5	5	0	731.8	-	-	0.5	5	5	5	743.0	10.9	-
160	3	1	0.2	5	5	0	3372.0	-	-	0.3	5	5	1	3394.8	2.7	-	1.6	5	5	5	3371.0	3.1	-
160	3	2	0.1	5	5	1	1726.8	5.9	-	0.1	5	5	1	1710.2	4.5	-	0.8	5	5	5	1726.8	5.2	-
160	3	4	0.0	5	5	0	957.8	-	-	0.0	5	5	1	951.8	8.3	-	0.8	5	5	5	957.8	8.9	-
160	3	8	0.0	5	5	0	548.4	-	-	0.0	5	5	0	532.0	-	-	0.7	5	5	5	548.4	14.8	-

Table 4.2: Problem $V, Q|sd, u, r_i, d_i|\sum C_i$ (part 1)

n V Q	w-t							w-w							w						
	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$
20 1 1	0.0	5	5	2	240.2	8.7	-	0.0	5	5	1	222.6	10.0	-	0.0	5	5	5	240.0	7.8	-
20 1 2	0.0	5	5	1	134.8	4.0	-	0.0	5	5	1	118.0	9.0	-	0.0	5	5	5	133.8	9.8	-
20 1 4	0.0	5	5	1	70.0	22.0	-	0.0	5	5	1	69.4	15.4	-	0.0	5	5	5	70.0	16.7	-
20 1 8	0.0	5	5	0	44.4	-	-	0.0	5	5	0	45.2	-	-	0.0	5	5	5	44.4	22.5	-
20 2 1	0.0	5	5	1	119.2	12.2	-	0.0	5	5	4	119.8	10.1	-	0.0	5	5	5	119.2	10.1	-
20 2 2	0.0	5	5	1	69.6	15.9	-	0.0	5	5	3	64.2	19.6	-	0.0	5	5	5	69.6	17.1	-
20 2 4	0.0	5	5	0	47.6	-	-	0.0	5	5	0	47.0	-	-	0.0	5	5	5	47.6	22.3	-
20 2 8	0.0	5	5	0	33.4	-	-	0.0	5	5	0	32.0	-	-	0.0	5	5	5	33.4	30.3	-
20 3 1	0.0	5	5	1	83.8	16.7	-	0.0	5	5	1	90.0	17.6	-	0.0	5	5	5	83.8	15.0	-
20 3 2	0.0	5	5	0	55.0	-	-	0.0	5	5	1	53.2	15.8	-	0.0	5	5	5	55.0	18.5	-
20 3 4	0.0	5	5	0	35.6	-	-	0.0	5	5	0	35.4	-	-	0.0	5	5	5	35.6	27.8	-
20 3 8	0.0	5	5	0	28.6	-	-	0.0	5	5	0	28.6	-	-	0.0	5	5	5	28.6	36.1	-
40 1 1	0.0	5	5	0	775.2	-	-	0.0	5	5	0	863.6	-	-	0.0	5	5	5	775.2	4.2	-
40 1 2	0.0	5	5	0	461.2	-	-	0.0	5	5	1	446.2	6.8	-	0.0	5	5	5	461.2	5.3	-
40 1 4	0.0	5	5	1	237.6	7.6	-	0.0	5	5	2	256.4	8.7	-	0.0	5	5	5	237.6	9.6	-
40 1 8	0.0	5	5	0	137.4	-	-	0.0	5	5	1	139.0	16.1	-	0.0	5	5	5	137.4	17.0	-
40 2 1	0.0	5	5	1	436.4	4.1	-	0.0	5	5	1	432.0	4.3	-	0.1	5	5	5	436.4	5.7	-
40 2 2	0.0	5	5	0	238.4	-	-	0.0	5	5	2	236.6	9.2	-	0.0	5	5	5	238.4	8.9	-
40 2 4	0.0	5	5	2	132.4	18.4	-	0.0	5	5	1	141.2	13.4	-	0.0	5	5	5	132.4	15.3	-
40 2 8	0.0	5	5	0	86.4	-	-	0.0	5	5	0	88.8	-	-	0.0	5	5	5	86.4	21.4	-
40 3 1	0.0	5	5	1	313.6	7.0	-	0.0	5	5	1	300.8	6.8	-	0.1	5	5	5	313.6	7.8	-
40 3 2	0.0	5	5	0	177.2	-	-	0.0	5	5	1	169.8	9.6	-	0.0	5	5	5	177.2	12.3	-
40 3 4	0.0	5	5	0	104.8	-	-	0.0	5	5	1	104.8	22.0	-	0.0	5	5	5	104.8	18.6	-
40 3 8	0.0	5	5	0	71.8	-	-	0.0	5	5	0	71.6	-	-	0.0	5	5	5	71.8	26.3	-
80 1 1	0.0	5	5	2	3113.8	2.2	-	0.1	5	5	4	3276.8	2.2	-	0.1	5	5	5	3113.2	2.1	-
80 1 2	0.0	5	5	1	1736.8	2.6	-	0.0	5	5	4	1633.4	3.3	-	0.1	5	5	5	1736.6	2.9	-
80 1 4	0.0	5	5	0	869.4	-	-	0.0	5	5	1	878.8	3.6	-	0.1	5	5	5	869.4	5.4	-
80 1 8	0.0	5	5	0	462.0	-	-	0.0	5	5	1	458.4	8.0	-	0.1	5	5	5	462.0	10.4	-
80 2 1	0.0	5	5	1	1769.8	3.2	-	0.1	5	5	2	1645.2	3.0	-	0.2	5	5	5	1769.8	2.9	-
80 2 2	0.0	5	5	1	892.2	6.1	-	0.0	5	5	2	841.8	5.9	-	0.1	5	5	5	892.2	5.5	-
80 2 4	0.0	5	5	0	440.4	-	-	0.0	5	5	0	461.8	-	-	0.1	5	5	5	440.4	11.6	-
80 2 8	0.0	5	5	0	270.8	-	-	0.0	5	5	1	266.4	19.9	-	0.1	5	5	5	270.8	17.4	-
80 3 1	0.0	5	5	0	1155.6	-	-	0.0	5	5	2	1158.2	4.7	-	0.3	5	5	5	1155.4	4.1	-
80 3 2	0.0	5	5	1	603.8	7.4	-	0.0	5	5	2	594.8	8.5	-	0.2	5	5	5	603.8	7.7	-
80 3 4	0.0	5	5	1	337.8	13.5	-	0.0	5	5	0	332.8	-	-	0.1	5	5	5	337.8	14.0	-
80 3 8	0.0	5	5	0	216.6	-	-	0.0	5	5	0	215.8	-	-	0.1	5	5	5	216.6	19.5	-
160 1 1	0.2	5	5	1	12987.4	0.8	-	0.3	5	5	2	12605.8	1.0	-	0.4	5	5	5	12984.0	0.8	-
160 1 2	0.1	5	5	0	6457.4	-	-	0.1	5	5	1	6535.2	1.5	-	0.3	5	5	5	6456.8	1.4	-
160 1 4	0.0	5	5	0	3294.0	-	-	0.0	5	5	3	3490.6	2.3	-	0.2	5	5	5	3293.6	2.6	-
160 1 8	0.0	5	5	0	1749.0	-	-	0.0	5	5	1	1737.8	5.4	-	0.2	5	5	5	1749.0	4.9	-
160 2 1	0.2	5	5	0	6649.4	-	-	0.2	5	5	2	6758.0	1.5	-	0.8	5	5	5	6649.4	1.5	-
160 2 2	0.1	5	5	1	3379.4	2.3	-	0.1	5	5	1	3396.0	2.9	-	0.5	5	5	5	3379.4	2.4	-
160 2 4	0.0	5	5	1	1757.8	4.7	-	0.0	5	5	0	1776.4	-	-	0.4	5	5	5	1757.8	4.7	-
160 2 8	0.0	5	5	0	937.2	-	-	0.0	5	5	1	925.8	10.9	-	0.4	5	5	5	937.2	9.3	-
160 3 1	0.2	5	5	0	4274.2	-	-	0.2	5	5	2	4338.4	1.9	-	1.2	5	5	5	4274.0	2.0	-
160 3 2	0.1	5	5	0	2278.8	-	-	0.1	5	5	2	2376.8	3.3	-	0.8	5	5	5	2278.8	3.7	-
160 3 4	0.0	5	5	0	1232.4	-	-	0.0	5	5	0	1212.4	-	-	0.7	5	5	5	1232.4	7.0	-
160 3 8	0.0	5	5	0	699.0	-	-	0.0	5	5	0	678.8	-	-	0.7	5	5	5	699.0	11.4	-

Table 4.3: Problem $V, Q|sd, u, r_i, d_i|\sum C_i$ (part 2)

The formulation is still extremely effective: all instances for which a feasible solution exists are solved to optimality in less than one second, with few exceptions. The greedy algorithm instead struggles in finding feasible solutions, especially for the case t-t. Also, when a feasible solution is found, the gap with respect to the optimal

solution is high. This is due to the fact that the solution space is narrow so the wrong choices made by the greedy algorithm have a strong impact. Anyway, given the excellent performance of the formulation, there seems to be no need to refine the greedy heuristic to get satisfactory solutions.

4.6.4 Computational results for $V, Q|sd, r_i, d_i|\sum C_i$

Tables 4.5 and 4.6 report results for problem $V, Q|sd, r_i, d_i|\sum C_i$, i.e, the case where demands are not forced to be unitary. Columns are the same as in former sections. In addition, the case with no release dates and due dates is reported in Table 4.4.

n	V	Q	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$	n	V	Q	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$
20	1	1	0.0	5	5	5	114.4	6.3	-	80	1	1	0.4	5	5	5	1404.2	4.8	-
20	1	2	0.1	5	5	5	86.0	8.2	-	80	1	2	509.9	5	4	5	987.8	10.0	5.6
20	1	4	0.1	5	5	5	73.4	17.5	-	80	1	4	1157.7	5	2	5	902.0	10.8	13.6
20	1	8	0.1	5	5	5	69.0	15.5	-	80	1	8	1801.3	5	0	5	-	-	11.9
20	2	1	0.0	5	5	5	66.6	5.4	-	80	2	1	0.4	5	5	5	733.2	4.8	-
20	2	2	0.1	5	5	5	53.8	8.8	-	80	2	2	1093.1	5	2	5	527.5	10.0	8.2
20	2	4	0.1	5	5	5	50.2	10.2	-	80	2	4	1345.9	5	2	5	472.5	10.0	11.6
20	2	8	0.1	5	5	5	44.6	10.9	-	80	2	8	1801.1	5	0	5	-	-	16.5
20	3	1	0.0	5	5	5	51.0	4.7	-	80	3	1	0.4	5	5	5	510.0	4.9	-
20	3	2	0.0	5	5	5	42.2	9.3	-	80	3	2	384.8	5	4	5	361.8	9.3	11.2
20	3	4	0.0	5	5	5	38.2	9.9	-	80	3	4	550.9	5	4	5	303.8	12.3	16.4
20	3	8	0.0	5	5	5	36.4	9.3	-	80	3	8	1801.4	5	0	5	-	-	11.7
40	1	1	0.1	5	5	5	404.0	5.6	-	160	1	1	3.2	5	5	5	5014.2	1.4	-
40	1	2	0.6	5	5	5	283.2	11.1	-	160	1	2	1801.0	5	0	5	-	-	7.0
40	1	4	1.9	5	5	5	249.6	10.4	-	160	1	4	1800.5	5	0	5	-	-	5.6
40	1	8	3.8	5	5	5	209.0	14.8	-	160	1	8	1800.3	5	0	5	-	-	6.2
40	2	1	0.1	5	5	5	216.2	5.4	-	160	2	1	3.4	5	5	5	2565.0	1.4	-
40	2	2	0.8	5	5	5	157.0	10.1	-	160	2	2	1448.0	5	1	5	1760.0	9.9	5.5
40	2	4	0.8	5	5	5	134.4	11.2	-	160	2	4	1800.4	5	0	5	-	-	6.6
40	2	8	3.2	5	5	5	135.0	11.2	-	160	2	8	1800.5	5	0	5	-	-	5.9
40	3	1	0.1	5	5	5	154.8	5.2	-	160	3	1	2.5	5	5	5	1749.6	1.4	-
40	3	2	0.3	5	5	5	120.4	8.5	-	160	3	2	1801.4	5	0	5	-	-	5.8
40	3	4	5.9	5	5	5	105.0	11.3	-	160	3	4	1800.7	5	0	5	-	-	7.0
40	3	8	1.0	5	5	5	90.6	12.4	-	160	3	8	1800.8	5	0	5	-	-	6.5

Table 4.4: Problem $V, Q|sd|\sum C_i$

In this case that the performance of the formulation deteriorates: the computing time sharply increases for instances with at least 80 requests and some of them are not solved to optimality. Thus, the variance of customer demands makes the problem more difficult and the difficulty increases with vehicle capacity. Still, the greedy algorithm struggles in finding good quality solutions, with the average error being often above 10%, even on instances not solved to optimality.

A similar trend as in Table 4.4 is observed in tables 4.5 and 4.6. Computing times decrease with respect to Table 4.4 thanks to the narrower solution space but there

n	V	Q	t-t							t-w							t						
			CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$
20	1	1	0.0	0	0	0	-	-	-	0.0	1	1	0	142.0	-	-	0.0	5	5	5	139.6	13.4	-
20	1	2	0.0	0	0	0	-	-	-	0.0	4	4	0	119.0	-	-	0.0	5	5	5	121.4	19.1	-
20	1	4	0.0	1	1	1	91.0	14.3	-	0.0	4	4	0	112.0	-	-	0.0	5	5	5	120.8	15.5	-
20	1	8	0.0	2	2	0	109.5	-	-	0.0	3	3	1	109.7	17.0	-	0.0	5	5	5	115.8	15.3	-
20	2	1	0.0	0	0	0	-	-	-	0.0	1	1	0	79.0	-	-	0.0	5	5	5	77.4	20.5	-
20	2	2	0.0	1	1	0	68.0	-	-	0.0	4	4	0	67.0	-	-	0.0	5	5	5	65.2	23.5	-
20	2	4	0.0	2	2	0	70.5	-	-	0.0	3	3	0	60.3	-	-	0.0	5	5	5	66.2	24.1	-
20	2	8	0.0	4	4	0	65.3	-	-	0.0	5	5	0	64.0	-	-	0.0	5	5	5	64.6	24.6	-
20	3	1	0.0	0	0	0	-	-	-	0.0	1	1	0	51.0	-	-	0.0	5	5	5	58.0	20.9	-
20	3	2	0.0	3	3	0	51.3	-	-	0.0	5	5	0	49.2	-	-	0.0	5	5	5	50.8	22.8	-
20	3	4	0.0	2	2	0	47.5	-	-	0.0	5	5	0	47.4	-	-	0.0	5	5	5	46.0	30.2	-
20	3	8	0.0	4	4	0	46.5	-	-	0.0	5	5	1	50.4	29.5	-	0.0	5	5	5	47.4	21.3	-
40	1	1	0.0	0	0	0	-	-	-	0.0	1	1	0	542.0	-	-	0.1	5	5	5	513.2	10.1	-
40	1	2	0.0	1	1	0	433.0	-	-	0.1	3	3	0	416.0	-	-	0.3	5	5	5	409.4	11.3	-
40	1	4	0.1	3	3	0	387.3	-	-	0.1	3	3	0	375.3	-	-	0.2	5	5	5	391.4	13.6	-
40	1	8	0.0	4	4	0	381.8	-	-	0.1	4	4	0	378.8	-	-	0.2	5	5	5	392.8	11.6	-
40	2	1	0.0	0	0	0	-	-	-	0.0	2	2	0	260.0	-	-	0.1	5	5	5	267.4	11.5	-
40	2	2	0.0	1	1	0	195.0	-	-	0.1	4	4	0	217.5	-	-	0.2	5	5	5	218.0	15.9	-
40	2	4	0.1	4	4	0	201.5	-	-	0.1	5	5	0	214.6	-	-	0.3	5	5	5	200.8	14.2	-
40	2	8	0.1	4	4	0	226.5	-	-	0.1	5	5	0	217.6	-	-	0.2	5	5	5	226.0	13.8	-
40	3	1	0.0	0	0	0	-	-	-	0.0	4	4	0	185.3	-	-	0.1	5	5	5	189.0	13.2	-
40	3	2	0.1	3	3	0	159.0	-	-	0.1	4	4	0	160.0	-	-	0.4	5	5	5	156.4	16.7	-
40	3	4	0.1	4	4	1	144.8	11.9	-	0.1	4	4	0	157.3	-	-	0.2	5	5	5	148.8	18.5	-
40	3	8	0.1	4	4	0	152.5	-	-	0.1	5	5	0	144.6	-	-	0.3	5	5	5	150.4	16.1	-
80	1	1	0.0	0	0	0	-	-	-	0.2	4	4	0	1813.5	-	-	0.7	5	5	5	1759.8	8.0	-
80	1	2	0.8	3	3	0	1622.7	-	-	11.2	5	5	0	1500.0	-	-	6.2	5	5	5	1503.0	10.8	-
80	1	4	0.5	3	3	0	1440.3	-	-	4.0	5	5	1	1482.8	6.8	-	4.9	5	5	5	1420.2	9.1	-
80	1	8	0.5	5	5	0	1348.8	-	-	1.2	5	5	0	1411.4	-	-	3.3	5	5	5	1339.0	8.7	-
80	2	1	0.1	0	0	0	-	-	-	0.1	3	3	0	938.0	-	-	1.0	5	5	5	893.0	9.5	-
80	2	2	1.5	5	5	0	793.4	-	-	2.1	5	5	0	766.0	-	-	20.4	5	5	5	775.6	12.4	-
80	2	4	7.5	5	5	0	723.0	-	-	27.0	5	5	0	725.4	-	-	11.3	5	5	5	714.0	12.5	-
80	2	8	1.1	5	5	0	745.0	-	-	1.7	5	5	0	728.8	-	-	2.4	5	5	5	739.0	12.0	-
80	3	1	0.1	0	0	0	-	-	-	0.2	4	4	0	636.8	-	-	0.9	5	5	5	611.0	11.0	-
80	3	2	3.4	5	5	0	551.0	-	-	1.6	5	5	0	521.2	-	-	6.9	5	5	5	537.6	12.8	-
80	3	4	1.9	5	5	0	501.2	-	-	7.2	5	5	0	504.6	-	-	15.4	5	5	5	497.8	13.8	-
80	3	8	1.1	5	5	1	501.4	9.7	-	3.2	5	5	0	512.8	-	-	3.6	5	5	5	498.0	12.7	-
160	1	1	0.2	0	0	0	-	-	-	3.5	5	5	0	6628.6	-	-	8.5	5	5	5	6553.4	6.2	-
160	1	2	538.8	5	4	0	5484.0	-	-	559.8	5	4	0	5547.0	-	-	1028.3	5	3	5	5483.3	7.5	7.9
160	1	4	351.3	5	5	0	5284.2	-	-	504.6	5	4	0	5248.8	-	-	786.8	5	3	5	5407.7	5.8	5.3
160	1	8	64.1	5	5	0	5142.0	-	-	131.5	5	5	0	5157.2	-	-	37.7	5	5	5	5127.2	5.8	-
160	2	1	0.3	0	0	0	-	-	-	2.9	5	5	0	3299.8	-	-	14.9	5	5	5	3177.0	6.2	-
160	2	2	1451.8	5	1	0	2708.0	-	-	955.0	5	3	0	2864.3	-	-	1440.5	5	2	5	2828.5	8.1	8.1
160	2	4	602.8	5	4	0	2720.3	-	-	589.0	5	4	0	2648.5	-	-	1094.9	5	2	5	2739.5	7.0	7.5
160	2	8	365.8	5	4	0	2711.0	-	-	114.1	5	5	0	2705.0	-	-	370.9	5	4	5	2701.0	5.6	7.8
160	3	1	0.3	0	0	0	-	-	-	4.2	5	5	0	2241.6	-	-	9.2	5	5	5	2224.8	8.6	-
160	3	2	1537.2	5	1	0	1874.0	-	-	790.3	5	3	0	1885.7	-	-	981.5	5	3	5	1867.3	9.6	10.2
160	3	4	490.4	5	5	0	1878.2	-	-	1357.4	5	2	0	1845.0	-	-	513.1	5	5	5	1872.4	9.0	-
160	3	8	413.3	5	4	0	1813.3	-	-	917.9	5	3	0	1835.0	-	-	413.4	5	4	5	1809.5	9.1	10.9

Table 4.5: Problem $V, Q|sd, r_i, d_i|\sum C_i$ (part 1)

n V Q	w-t							w-w							w					
	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$	CPU	feas	opt	gr	$\sum C_i$	$gapGr^O$	$gapGr^{UB}$
20 1 1	0.0	0	0	0	-	-	-	0.0	2	2	0	141.5	-	-	0.0	5	5	5	145.4	18.7	-
20 1 2	0.0	2	2	1	145.5	9.9	-	0.0	4	4	0	144.8	-	-	0.0	5	5	5	143.8	14.3	-
20 1 4	0.0	2	2	0	115.0	-	-	0.0	1	1	0	161.0	-	-	0.0	5	5	5	130.4	12.6	-
20 1 8	0.0	2	2	0	121.5	-	-	0.0	4	4	1	128.8	16.5	-	0.0	5	5	5	133.4	10.6	-
20 2 1	0.0	0	0	0	-	-	-	0.0	3	3	0	77.3	-	-	0.0	5	5	5	93.4	11.3	-
20 2 2	0.0	4	4	0	77.8	-	-	0.0	4	4	1	75.3	22.5	-	0.0	5	5	5	74.4	20.2	-
20 2 4	0.0	2	2	0	73.5	-	-	0.0	4	4	1	78.5	11.8	-	0.0	5	5	5	78.6	17.7	-
20 2 8	0.0	5	5	0	75.4	-	-	0.0	5	5	0	70.8	-	-	0.0	5	5	5	74.8	15.1	-
20 3 1	0.0	1	1	0	53.0	-	-	0.0	2	2	0	63.0	-	-	0.0	5	5	5	63.0	19.3	-
20 3 2	0.0	3	3	0	52.3	-	-	0.0	5	5	0	55.2	-	-	0.0	5	5	5	55.6	18.1	-
20 3 4	0.0	4	4	0	51.8	-	-	0.0	5	5	3	51.4	23.8	-	0.0	5	5	5	54.2	23.9	-
20 3 8	0.0	5	5	1	53.0	15.4	-	0.0	5	5	0	53.8	-	-	0.0	5	5	5	52.8	21.1	-
40 1 1	0.0	1	1	0	503.0	-	-	0.0	2	2	0	563.5	-	-	0.1	5	5	5	552.8	8.5	-
40 1 2	0.0	4	4	0	470.5	-	-	0.1	5	5	0	482.0	-	-	0.1	5	5	5	458.2	8.4	-
40 1 4	0.0	4	4	0	499.8	-	-	0.0	5	5	0	486.4	-	-	0.1	5	5	5	477.4	7.5	-
40 1 8	0.0	5	5	0	420.8	-	-	0.0	5	5	0	469.6	-	-	0.1	5	5	5	418.0	9.8	-
40 2 1	0.0	1	1	0	261.0	-	-	0.0	3	3	0	289.3	-	-	0.1	5	5	5	288.8	10.1	-
40 2 2	0.0	4	4	0	258.5	-	-	0.1	5	5	0	252.8	-	-	0.1	5	5	5	254.6	12.7	-
40 2 4	0.0	4	4	0	249.5	-	-	0.1	5	5	0	262.2	-	-	0.1	5	5	5	250.8	12.8	-
40 2 8	0.0	4	4	1	245.0	10.8	-	0.1	5	5	0	248.0	-	-	0.1	5	5	5	241.2	13.3	-
40 3 1	0.0	1	1	0	198.0	-	-	0.0	5	5	0	205.6	-	-	0.1	5	5	5	199.0	9.8	-
40 3 2	0.0	5	5	0	193.2	-	-	0.1	5	5	0	187.0	-	-	0.1	5	5	5	190.2	12.8	-
40 3 4	0.1	5	5	0	179.2	-	-	0.1	5	5	0	181.4	-	-	0.2	5	5	5	178.6	16.7	-
40 3 8	0.0	5	5	0	180.0	-	-	0.0	5	5	0	176.0	-	-	0.1	5	5	5	179.6	12.8	-
80 1 1	0.1	2	2	0	1881.5	-	-	0.2	4	4	0	1983.8	-	-	0.3	5	5	5	1911.0	7.6	-
80 1 2	0.2	5	5	0	1794.8	-	-	0.6	5	5	0	1765.6	-	-	0.6	5	5	5	1780.2	6.1	-
80 1 4	0.4	5	5	0	1750.4	-	-	0.3	5	5	0	1757.2	-	-	0.8	5	5	5	1738.0	6.8	-
80 1 8	0.2	5	5	0	1726.8	-	-	0.4	5	5	0	1773.2	-	-	0.4	5	5	5	1722.4	4.7	-
80 2 1	0.1	4	4	0	976.5	-	-	0.1	5	5	0	1038.8	-	-	0.8	5	5	5	956.8	9.5	-
80 2 2	1.2	5	5	0	935.8	-	-	0.6	5	5	1	894.0	5.0	-	1.9	5	5	5	929.8	9.5	-
80 2 4	0.3	5	5	1	908.6	4.3	-	0.4	5	5	0	880.8	-	-	0.9	5	5	5	906.0	7.2	-
80 2 8	0.3	5	5	0	911.2	-	-	0.2	5	5	0	901.2	-	-	0.7	5	5	5	910.2	6.5	-
80 3 1	0.1	2	2	0	660.0	-	-	0.1	5	5	0	700.6	-	-	0.6	5	5	5	668.6	12.1	-
80 3 2	1.0	5	5	0	634.4	-	-	0.5	5	5	0	634.2	-	-	2.2	5	5	5	633.2	9.5	-
80 3 4	0.3	5	5	0	651.4	-	-	0.3	5	5	0	646.0	-	-	1.3	5	5	5	650.8	8.6	-
80 3 8	0.2	5	5	0	633.6	-	-	0.2	5	5	0	645.4	-	-	0.8	5	5	5	633.0	8.8	-
160 1 1	1.7	5	5	0	7328.8	-	-	2.3	5	5	0	7433.8	-	-	3.4	5	5	5	7261.6	6.0	-
160 1 2	5.6	5	5	0	6831.8	-	-	15.0	5	5	0	6718.2	-	-	12.6	5	5	5	6825.4	3.9	-
160 1 4	1.8	5	5	0	6640.8	-	-	3.6	5	5	0	6537.2	-	-	2.9	5	5	5	6633.0	3.2	-
160 1 8	0.8	5	5	0	6540.6	-	-	1.4	5	5	0	6569.4	-	-	2.1	5	5	5	6535.0	2.7	-
160 2 1	2.3	5	5	0	3719.8	-	-	2.2	5	5	0	3756.0	-	-	12.3	5	5	5	3697.2	6.2	-
160 2 2	2.7	5	5	0	3485.4	-	-	9.9	5	5	0	3452.8	-	-	4.9	5	5	5	3482.8	4.6	-
160 2 4	2.8	5	5	0	3561.0	-	-	2.6	5	5	0	3459.0	-	-	6.6	5	5	5	3558.4	3.7	-
160 2 8	1.6	5	5	0	3337.4	-	-	2.1	5	5	1	3343.0	2.7	-	5.7	5	5	5	3336.2	3.7	-
160 3 1	2.0	5	5	0	2465.2	-	-	1.3	5	5	0	2516.0	-	-	12.7	5	5	5	2450.0	7.6	-
160 3 2	16.5	5	5	0	2305.4	-	-	19.6	5	5	0	2409.2	-	-	24.4	5	5	5	2303.0	6.5	-
160 3 4	2.2	5	5	0	2337.8	-	-	2.2	5	5	0	2296.2	-	-	6.4	5	5	5	2336.6	5.0	-
160 3 8	1.3	5	5	0	2297.0	-	-	1.0	5	5	1	2295.6	5.3	-	6.5	5	5	5	2296.8	4.9	-

Table 4.6: Problem $V, Q|sd, r_i, d_i|\sum C_i$ (part 2)

are still instances which are not solved to optimality. However, the formulation is always able to provide a feasible solution when it exists.

The following two sections contain some unsuccessful attempts carried out in order to find optimal solutions for the most difficult instances. For this reason they are presented after the computational results.

4.7 A surrogate relaxation of formulation (4.2)-(4.5)

In this section a surrogate relaxation of formulation (4.2)-(4.5) is proposed. We introduce variables $x_{ik} = \sum_{v=1}^{V} x_{ikv}$ and we aggregate capacity constraints (4.3) on all the vehicles to obtain model:

$$\min \sum_{i \in \mathcal{R}} \sum_{k \in \mathcal{K}_i} c_{ik} x_{ik} \tag{4.6}$$

s.t.:

$$\sum_{\{i \in \mathcal{R}[j,j+1], s_i \leq j\}} q_i x_{ik} + \sum_{\{i \in \mathcal{R}[j,j+1], s_i > j+1\}} q_i x_{ik-1} \leq QV \tag{4.7}$$

$$\sum_{k \in \mathcal{K}_i} x_{ik} = 1 \tag{4.8}$$

$$x_{ik} \in \{0,1\} \tag{4.9}$$

with $1 \leq i \leq n, 0 \leq j \leq m-1, 1 \leq k \leq K$.

In this model, capacity constraints (4.7) ensure that the global capacity offered by the vehicle fleet is enough to satisfy the requests assigned in each tour to interval $[j, j+1]$, but they do not guarantee that these requests can be distributed between the V vehicles, satisfying individual vehicle capacity. The value of the relaxation provides a valid lower bound.

The potential usefulness of the relaxation relies on the ability to solve it quickly. Unfortunately, experiments conducted on the hardest instances show that, when the optimal solution can not be found with model (4.2)-(4.5) within the time limit, this happens also for formulation (4.6)-(4.9).

In order to overcome this difficulty, the formulation was reinforced. Let $\mathcal{R}_\beta[j, j+1]$ denote the set of requests that cover segment $[j, j+1]$ with a demand $q_i \geq \beta$, for each $2 \leq \beta \leq Q$. Then, the following inequalities hold:

$$\sum_{\{i \in \mathcal{R}_\beta[j,j+1], s_i \leq j\}} x_{ik} + \sum_{\{i \in \mathcal{R}_\beta[j,j+1], s_i > j+1\}} x_{ik-1} \leq \left\lfloor \frac{Q}{\beta} \right\rfloor V \tag{4.10}$$

with $0 \leq j \leq m-1, 1 \leq k \leq K, 2 \leq \beta \leq Q$.

These inequalities easily derive from Constraints (4.3). Indeed, by dividing Constraints (4.3) by β, it is easy to see that

$$\sum_{\{i \in \mathcal{R}_\beta[j,j+1], s_i \leq j\}} x_{ikv} + \sum_{\{i \in \mathcal{R}_\beta[j,j+1], s_i > j+1\}} x_{ik-1v} \leq \frac{Q}{\beta}$$

with $0 \leq j \leq m-1, 1 \leq k \leq K, 1 \leq v \leq V$.

Then, since the left-hand side is integer, we can apply the floor operator to the right-hand side (which keeps the inequality valid) and sum the constraints on the V vehicles to obtain Constraints (4.10). Unfortunately, experiments did not show any acceleration on computing times when these valid inequalities were added.

Table 4.7 reports results of the solution of formulations (4.2)-(4.5) and (4.6)-(4.9) on the instances of problem $V, Q|sd|\sum C_i$. The computing environment is the same as the one used in the experiments presented in section 4.6. All results are averaged over the 5 instances with the same value of n, V and Q. Column *value* and column *opt* report, respectively, the value and the number of optimal solutions to formulation (4.2)-(4.5). Column CPU reports the solution time of formulation (4.2)-(4.5). Column gap_{sr} reports the percentage gap between the solution value of formulation (4.2)-(4.5) and the solution value of formulation (4.6)-(4.9). Column opt_{sr} reports the number of optimal solutions calculated by solving formulation (4.6)-(4.9). Column $gap_{sr,vi}$ reports the same information as column gap_{sr} but with inequalities (4.10) in formulation (4.6)-(4.9). Column $opt_{sr,vi}$ reports the number of optimal solutions calculated by solving formulation (4.6)-(4.9) with inequalities (4.10). Columns gap_{sr} and $gap_{sr,vi}$ are calculated as $\frac{x_{or}-x_{sr}}{x_{or}}$ and $\frac{x_{sr}-x_{sr,vi}}{x_{sr}}$ respectively, where x_{or} denotes the solution value of formulation (4.2)-(4.5), x_{sr} the solution value of formulation (4.6)-(4.9), and $x_{sr,vi}$ denote the solution value of formulation (4.6)-(4.9) with inequalities (4.10). Columns CPU_{sr} and $CPU_{sr,vi}$ report the solution time of formulation (4.6)-(4.9) without and with inequalities (4.10), respectively.

Table 4.7 shows that formulation (4.6)-(4.9) is as hard to solve as formulation (4.2)-(4.5), even with inequalities (4.10). In fact, the number of optimal solutions calculated is almost the same, as well as the solution time.

4.7.1 Feasibility of the solutions

Tables 4.8 and 4.9 report, for every combination of the parameters, the value and the number of solutions to formulation (4.6)-(4.9) that are feasible for problem $V, Q|sd|\sum C_i$. Two different limits were set in CPLEX to solve the formulation: 10 seconds and 5 minutes, which refer to tables 4.8 and 4.9, respectively.

n	V	Q	$value$	opt	CPU	gap_{sr}	opt_{sr}	CPU_{sr}	$gap_{sr,vi}$	$opt_{sr,vi}$	$CPU_{sr,vi}$
20	1	1	114.4	5	0.0	0.0	5	0.0	0.0	5	0.0
20	1	2	86.0	5	0.1	0.0	5	0.1	0.0	5	0.1
20	1	4	73.4	5	0.1	0.0	5	0.1	0.0	5	0.1
20	1	8	69.0	5	0.1	0.0	5	0.1	0.0	5	0.1
20	2	1	66.6	5	0.0	0.0	5	0.0	0.0	5	0.0
20	2	2	53.8	5	0.1	0.0	5	0.1	0.0	5	0.1
20	2	4	50.2	5	0.1	2.5	5	0.0	2.5	5	0.1
20	2	8	44.6	5	0.1	3.2	5	0.0	2.7	5	0.0
20	3	1	51.0	5	0.0	0.0	5	0.0	0.0	5	0.0
20	3	2	42.2	5	0.0	0.0	5	0.0	0.0	5	0.0
20	3	4	38.2	5	0.0	0.0	5	0.0	0.0	5	0.0
20	3	8	36.4	5	0.0	1.4	5	0.0	1.4	5	0.0
40	1	1	404.0	5	0.1	0.0	5	0.1	0.0	5	0.1
40	1	2	283.2	5	0.8	0.0	5	1.4	0.0	5	1.4
40	1	4	249.6	5	2.2	0.0	5	3.2	0.0	5	3.6
40	1	8	209.0	5	4.3	0.0	5	6.9	0.0	5	9.0
40	2	1	216.2	5	0.1	0.0	5	0.0	0.0	5	0.0
40	2	2	157.0	5	0.8	0.1	5	0.1	0.1	5	0.1
40	2	4	134.4	5	0.8	1.7	5	0.6	1.7	5	0.8
40	2	8	135.0	5	3.2	3.6	5	10.1	3.3	5	6.9
40	3	1	154.8	5	0.1	0.0	5	0.0	0.0	5	0.0
40	3	2	120.4	5	0.3	0.0	5	0.1	0.0	5	0.1
40	3	4	105.0	5	5.8	2.8	5	0.3	2.3	5	0.4
40	3	8	90.6	5	1.0	1.8	5	0.2	1.3	5	0.2
80	1	1	1404.2	5	0.4	0.0	5	0.5	0.0	5	0.5
80	1	2	982.2	4	509.8	0.0	3	775.3	0.0	3	775.3
80	1	4	829.2	2	1158.0	−0.2	2	1342.5	−0.3	2	1282.1
80	1	8	722.6	0	1801.2	−0.1	0	1801.3	−0.1	0	1801.3
80	2	1	733.2	5	0.4	0.0	5	0.1	0.0	5	0.1
80	2	2	531.8	2	1093.3	0.1	3	806.5	0.1	3	805.3
80	2	4	449.2	2	1343.6	2.3	1	1463.6	2.1	1	1483.5
80	2	8	395.6	0	1801.3	2.7	0	1802.0	2.3	0	1802.1
80	3	1	510.0	5	0.4	0.0	5	0.1	0.0	5	0.1
80	3	2	364.6	4	385.0	0.1	5	105.8	0.1	5	104.9
80	3	4	312.4	4	560.6	2.3	4	410.9	2.0	5	118.3
80	3	8	295.4	0	1801.6	3.3	1	1624.3	2.7	1	1475.7
160	1	1	5014.2	5	3.4	0.0	5	3.7	0.0	5	3.7
160	1	2	3469.8	0	1801.0	−0.6	0	1800.5	−0.6	0	1800.6
160	1	4	2867.6	0	1800.6	0.2	0	1800.7	0.3	0	1800.6
160	1	8	2585.8	0	1800.4	0.2	0	1800.4	−1.2	0	1801.3
160	2	1	2565.0	5	3.7	0.0	5	1.1	0.0	5	1.1
160	2	2	1794.4	1	1448.9	0.1	1	1525.0	0.1	1	1525.4
160	2	4	1480.0	0	1800.6	2.0	0	1801.8	2.6	0	1801.4
160	2	8	1320.0	0	1802.2	3.4	0	1801.7	2.3	0	1802.0
160	3	1	1749.6	5	2.7	0.0	5	0.6	0.0	5	0.6
160	3	2	1243.8	0	1801.3	0.2	0	1801.9	0.2	0	1801.9
160	3	4	1077.2	0	1800.9	3.7	0	1802.7	3.4	0	1802.2
160	3	8	983.2	0	1800.6	5.7	0	1803.0	5.1	0	1802.2

Table 4.7: Problem $V, Q|sd|\sum C_i$

n	V	Q	CPU	$value$	# feasible
20	1	1	0.0	114.4	5
20	1	2	0.1	86.0	5
20	1	4	0.1	73.4	5
20	1	8	0.1	69.0	5
20	2	1	0.0	66.6	5
20	2	2	0.0	53.8	4
20	2	4	0.0	49.0	1
20	2	8	0.0	43.2	0
20	3	1	0.0	51.0	5
20	3	2	0.0	42.2	3
20	3	4	0.0	38.2	1
20	3	8	0.0	35.8	0
40	1	1	0.1	404.0	5
40	1	2	0.6	283.2	5
40	1	4	1.9	249.6	5
40	1	8	4.1	209.0	5
40	2	1	0.0	216.2	5
40	2	2	0.1	156.8	0
40	2	4	0.2	131.8	0
40	2	8	3.4	130.2	0
40	3	1	0.0	154.8	5
40	3	2	0.1	120.4	1
40	3	4	0.1	101.8	0
40	3	8	0.1	89.0	0

n	V	Q	CPU	$value$	# feasible
80	1	1	0.4	1404.2	5
80	1	2	8.5	983.4	5
80	1	4	10.0	853.4	5
80	1	8	10.0	740.2	5
80	2	1	0.2	733.2	5
80	2	2	5.8	531.6	1
80	2	4	10.0	445.2	0
80	2	8	10.0	394.4	0
80	3	1	0.1	510.0	5
80	3	2	1.0	364.2	0
80	3	4	8.8	306.0	0
80	3	8	10.0	288.6	0
160	1	1	3.1	5014.2	5
160	1	2	10.0	3681.2	5
160	1	4	10.0	3075.0	5
160	1	8	10.0	2765.4	5
160	2	1	1.4	2565.0	5
160	2	2	8.8	1826.0	0
160	2	4	10.0	1524.2	0
160	2	8	10.0	1339.6	0
160	3	1	1.2	1749.6	5
160	3	2	10.0	1255.8	0
160	3	4	10.0	1064.2	0
160	3	8	10.0	959.0	0

Table 4.8: Problem $V, Q|sd|\sum C_i$ - 10 seconds time limit

n	V	Q	CPU	$value$	# feasible
20	1	1	0.0	114.4	5
20	1	2	0.1	86.0	5
20	1	4	0.1	73.4	5
20	1	8	0.1	69.0	5
20	2	1	0.0	66.6	5
20	2	2	0.0	53.8	4
20	2	4	0.0	49.0	1
20	2	8	0.0	43.2	0
20	3	1	0.0	51.0	5
20	3	2	0.0	42.2	3
20	3	4	0.0	38.2	1
20	3	8	0.0	35.8	0
40	1	1	0.1	404.0	5
40	1	2	0.6	283.2	5
40	1	4	1.9	249.6	5
40	1	8	4.0	209.0	5
40	2	1	0.0	216.2	5
40	2	2	0.1	156.8	0
40	2	4	0.2	131.8	0
40	2	8	4.4	130.2	0
40	3	1	0.0	154.8	5
40	3	2	0.1	120.4	1
40	3	4	0.1	101.8	0
40	3	8	0.1	89.0	0
80	1	1	0.4	1404.2	5
80	1	2	141.7	982.2	5
80	1	4	248.6	833.8	5
80	1	8	300.1	728.4	5
80	2	1	0.2	733.2	5
80	2	2	63.9	531.4	0
80	2	4	250.7	439.0	0
80	2	8	300.6	386.4	0
80	3	1	0.1	510.0	5
80	3	2	1.0	364.2	0
80	3	4	73.6	305.0	0
80	3	8	255.7	285.8	0
160	1	1	3.1	5014.2	5
160	1	2	300.1	3492.0	5
160	1	4	300.2	2917.4	5
160	1	8	300.1	2660.2	5
160	2	1	1.4	2565.0	5
160	2	2	240.9	1795.6	0
160	2	4	300.1	1457.6	0
160	2	8	300.1	1285.2	0
160	3	1	1.3	1749.6	5
160	3	2	245.6	1241.4	0
160	3	4	300.3	1042.8	0
160	3	8	300.5	933.0	0

Table 4.9: Problem $V, Q|sd|\sum C_i$ - 5 minutes time limit

4.7.2 Repairing the solutions of the surrogate relaxation

Formulation (4.6)-(4.9) provides solutions to problem $V, Q|sd, r_i, d_i|\sum C_i$ that may be infeasible when the demands are not unitary and $Q, V > 1$. In this section it is addressed the problem of trying to make such solutions feasible through a greedy heuristic and the performances of formulation (4.6)-(4.9) are compared with formulation (4.2)-(4.5).

Suppose formulation (4.6)-(4.9) has scheduled the requests in K different tours. For any tour $1 \leq k \leq K$, we assign the requests to vehicles by solving to optimality m Bin Packing Problems (BPP), one for each segment of the ring, where the bins are represented by the vehicles. If in tour k the BPP fails on segment $[j, j+1]$, we have to decide which requests must be assigned to tour $k+1$ among the ones that have starting station equal to j. We sort such requests according to the following criteria, taken hierarchically:

- greatest deadline first;
- shortest distance to reach the delivery node first;
- largest demand first;

Then we shift one request at a time starting from the first one in the ordering, until the BPP is solved.

In the BPP to be solved, every bin has a different capacity $c_i \leq q$. The reason is that when a subset $\mathcal{R}' \subset \mathcal{R}$ of requests is assigned to vehicle v on segment $[j, j+1]$, we have to record that choice and solve a BPP on segment $[j+1, j+2]$ where v has a capacity decreased by $\sum_{i \in \mathcal{R}'} q_i$.

Let u be the maximum number of available bins (vehicles) and assume that the potential bins are numbered as $1 \dots u$. Let c_i denote the capacity of bin i.

We introduce two types of binary decision variables:

$$y_i = \begin{cases} 1 & \text{if bin } i \text{ is used in the solution} \\ 0 & \text{otherwise.} \end{cases}$$

$$x_{ij} = \begin{cases} 1 & \text{if item } j \text{ is packed into bin } i \\ 0 & \text{otherwise.} \end{cases}$$

The integer linear program is then:

$$\min \sum_{i=1}^{u} y_i \tag{4.11}$$

s.t.:

$$\sum_{j=1}^{n} w_j x_{ij} \leq c_i y_i \qquad (1 \leq i \leq u) \tag{4.12}$$

$$\sum_{i=1}^{u} x_{ij} = 1 \qquad (1 \leq j \leq n) \tag{4.13}$$

$$x_{ij} \in \{0,1\} \qquad (1 \leq i \leq u,\ 1 \leq j \leq n) \tag{4.14}$$

$$y_i \in \{0,1\} \qquad (1 \leq i \leq u) \tag{4.15}$$

What follows is the pseudocode of the algorithm:

Algorithm 2 Greedy algorithm for the repairing phase

Input: assignment of requests for each tour $1 \leq k \leq K$

1: **for all** segments $(j, j+1)$ of the ring $(j = 0, ..., m-1)$ **do**
2: if a request i reaches its delivery station, increase the capacity of the vehicle serving i by q_i
3: try to solve a BPP with V bins each of capacity c_v and with requests covering the current segment
4: **while** the BPP has no solution **do**
5: create a list of the requests starting on segment $[j, j+1]$
6: sort the requests according to the criteria described above
7: move the first request in the ordering to tour $k+1$
8: **end while**
9: decrease the capacities of the vehicles by the sum of assigned requests
10: **end for**

Table 4.10 reports the results of Algorithm 2 on the instances of problem $V, Q|sd|\sum C_i$. Column *avg_value* reports the average value of the solutions to formulation (4.2)-(4.5). Column *avg_gap_gr* reports the average gap between the value of the solutions to formulation and the solution value calculated by the greedy algorithm described in section 4.6.1. The last four columns report the results of the solutions obtained by solving formulation (4.6)-(4.9) and then applying the repairing phase. Note that for problem $V, Q|sd|\sum C_i$, the first criterion (greatest deadline first) is never used to sort the requests to remove. In the repair phase tie-breaking criteria are applied in the two different orders 2,3 and 3,2. Columns *avg_gap_sdf* and *avg_gap_ldf* report such results, where *sdf* and *ldf* stand for *shortest distance first* and *largest demand first*.

					cf + repair					
					10 seconds		30 seconds		5 minutes	
n	V	Q	avg_value	avg_gap_gr	avg_gap_sdf	avg_gap_ldf	avg_gap_sdf	avg_gap_ldf	avg_gap_sdf	avg_gap_ldf
20	1	1	114.4	6.3	0.0	0.0	0.0	0.0	0.0	0.0
20	1	2	86.0	8.2	0.0	0.0	0.0	0.0	0.0	0.0
20	1	4	73.4	17.5	0.0	0.0	0.0	0.0	0.0	0.0
20	1	8	69.0	15.5	0.0	0.0	0.0	0.0	0.0	0.0
20	2	1	66.6	5.4	0.0	0.0	0.0	0.0	0.0	0.0
20	2	2	53.8	8.8	1.2	1.2	1.2	1.2	1.2	1.2
20	2	4	50.2	10.2	10.8	8.3	10.8	8.3	10.8	8.3
20	2	8	44.6	10.9	9.6	9.2	9.6	9.2	9.6	9.2
20	3	1	51.0	4.7	0.0	0.0	0.0	0.0	0.0	0.0
20	3	2	42.2	9.3	2.1	2.1	2.1	2.1	2.1	2.1
20	3	4	38.2	9.9	5.6	5.6	5.6	5.6	5.6	5.6
20	3	8	36.4	9.3	5.2	5.2	5.2	5.2	5.2	5.2
40	1	1	404.0	5.6	0.0	0.0	0.0	0.0	0.0	0.0
40	1	2	283.2	11.2	0.0	0.0	0.0	0.0	0.0	0.0
40	1	4	249.6	10.4	0.0	0.0	0.0	0.0	0.0	0.0
40	1	8	209.0	14.7	0.1	0.1	0.0	0.0	0.0	0.0
40	2	1	216.2	5.4	0.0	0.0	0.0	0.0	0.0	0.0
40	2	2	157.0	10.1	6.1	6.1	6.1	6.1	6.1	6.1
40	2	4	134.4	11.2	10.1	10.1	10.1	10.1	10.1	10.1
40	2	8	135.0	11.2	12.7	11.3	12.6	11.3	12.6	11.4
40	3	1	154.8	5.2	0.0	0.0	0.0	0.0	0.0	0.0
40	3	2	120.4	8.5	5.9	5.9	5.9	5.9	5.9	5.9
40	3	4	105.0	11.3	14.1	12.1	14.1	12.1	14.1	12.1
40	3	8	90.6	12.4	14.4	12.4	14.4	12.4	14.4	12.4
80	1	1	1404.2	4.8	0.0	0.0	0.0	0.0	0.0	0.0
80	1	2	982.2	9.1	0.1	0.2	0.0	0.1	0.0	0.0
80	1	4	829.4	12.5	5.5	5.0	2.3	3.0	0.8	0.7
80	1	8	722.6	11.9	3.4	5.3	2.1	2.6	0.9	1.0
80	2	1	733.2	4.8	0.0	0.0	0.0	0.0	0.0	0.0
80	2	2	531.6	9.0	5.8	4.6	4.4	4.4	5.8	5.8
80	2	4	449.2	11.0	16.7	15.4	15.1	15.0	15.6	14.4
80	2	8	394.6	16.8	20.4	20.4	20.0	18.7	17.4	17.8
80	3	1	510.0	4.9	0.0	0.0	0.0	0.0	0.0	0.0
80	3	2	364.6	9.7	5.7	5.7	5.7	5.7	5.7	5.7
80	3	4	312.4	13.2	16.6	14.4	15.1	15.3	16.7	15.4
80	3	8	294.4	12.1	16.6	15.4	16.5	14.5	18.6	16.5
160	1	1	5014.2	1.4	0.0	5.6	0.0	0.0	0.0	0.0
160	1	2	3469.0	7.1	15.3	15.3	1.8	3.9	0.6	0.7
160	1	4	2869.8	5.6	7.8	7.8	3.0	5.8	1.7	1.7
160	1	8	2582.4	6.1	9.6	9.6	7.1	7.1	3.0	2.8
160	2	1	2565.0	1.4	0.0	0.0	0.0	0.0	0.0	0.0
160	2	2	1796.6	6.3	9.1	9.1	6.4	6.2	6.2	5.1
160	2	4	1481.0	6.5	17.4	17.9	16.6	19.0	16.1	15.6
160	2	8	1329.2	5.2	20.9	20.1	18.3	18.7	19.9	18.3
160	3	1	1749.6	1.4	0.0	0.0	0.0	0.0	0.0	0.0
160	3	2	1244.2	5.9	7.6	7.9	4.5	5.0	3.8	3.8
160	3	4	1080.6	6.6	15.5	14.6	12.6	13.0	14.8	15.0
160	3	8	981.2	6.5	18.4	17.2	18.4	17.6	17.4	14.8

Table 4.10: Problem $V, Q|sd|\sum C_i$ - formulation (4.6)-(4.9) + Algorithm 2

Table 4.10 shows that on easy instances (20-40 requests) Algorithm 2 performs better than the greedy algorithm (described in section 4.6.1), and on hard instances (80-160 requests) the greedy algorithm performs better.

4.7.3 Adding a noise term in the objective function

To speed up the resolution of formulation (4.6)-(4.9) a symmetry-breaking mechanism was introduced. To this end, "noise" is added to the objective function. For every request i served in tour k the term $k\epsilon^i$ is added. The objective function is therefore modified as follows:

$$\min \sum_{i\in\mathcal{R}} \sum_{k\in\mathcal{K}_i} c_{ik}x_{ik} + \sum_{i\in\mathcal{R}} \sum_{k\in\mathcal{K}_i} k\epsilon^i x_{ik} \tag{4.16}$$

In order to solve the formulation on the instances of the problem $V, Q|sd|\sum C_i$, ϵ was set to 0.01. The noise is small enough to make sure that the optimal solution with the noise would also be optimal without the noise. In fact, if every request i is scheduled in a different tour, we have

$$\sum_{i=1}^{160}(161-i)0.01^i x_{ik} \approx 1.6$$

Table 4.11 reports the results of the solution of formulation (4.6)-(4.9) with the addition of the noise term solved on instances $V, Q|sd|\sum C_i$.

n	V	Q	CPU(s)	solved	feas	optimal	n	V	Q	CPU(s)	solved	feas	optimal
20	1	1	0.0	5	5	5	80	1	1	0.5	5	5	5
20	1	2	0.1	5	5	5	80	1	2	788.8	5	5	3
20	1	4	0.1	5	5	5	80	1	4	1225.2	5	5	2
20	1	8	0.1	5	5	5	80	1	8	1801.4	5	5	0
20	2	1	0.0	5	5	5	80	2	1	0.2	5	5	5
20	2	2	0.1	5	5	5	80	2	2	725.4	5	5	3
20	2	4	0.1	5	5	5	80	2	4	1542.8	5	5	1
20	2	8	0.1	5	5	5	80	2	8	1802.3	5	5	0
20	3	1	0.0	5	5	5	80	3	1	0.2	5	5	5
20	3	2	0.0	5	5	5	80	3	2	49.6	5	5	5
20	3	4	0.0	5	5	5	80	3	4	414.7	5	5	4
20	3	8	0.0	5	5	5	80	3	8	1573.9	5	5	1
40	1	1	0.1	5	5	5	160	1	1	3.4	5	5	5
40	1	2	1.3	5	5	5	160	1	2	1800.5	5	5	0
40	1	4	2.8	5	5	5	160	1	4	1800.6	5	5	0
40	1	8	7.2	5	5	5	160	1	8	1800.4	5	5	0
40	2	1	0.0	5	5	5	160	2	1	1.6	5	5	5
40	2	2	0.2	5	5	5	160	2	2	1668.5	5	5	1
40	2	4	0.7	5	5	5	160	2	4	1801.7	5	5	0
40	2	8	9.0	5	5	5	160	2	8	1801.7	5	5	0
40	3	1	0.0	5	5	5	160	3	1	1.0	5	5	5
40	3	2	0.1	5	5	5	160	3	2	1802.0	5	5	0
40	3	4	0.3	5	5	5	160	3	4	1802.8	5	5	0
40	3	8	0.2	5	5	5	160	3	8	1803.0	5	5	0

Table 4.11: Problem $V, Q|sd|\sum C_i$

If we compare the results with table 4.4 we see that there is no big difference in the number of instances solved to optimality between the two formulations.

4.8 A different surrogate relaxation

As an additional step, the model was further relaxed by defining a single capacity constraint per tour: the sum of the loads of the vehicles over all intervals $[j, j+1]$ is constrained to be less than QVm for each tour.

We start from formulation (4.2)-(4.5), that is reported here to ease the readability:

$$\min \sum_{i \in \mathcal{R}} \sum_{v=1}^{V} \sum_{k \in \mathcal{K}_i} c_{ik} x_{ikv} \tag{4.17}$$

s.t.:

$$\sum_{\{i \in \mathcal{R}[j,j+1], s_i \leq j\}} q_i x_{ikv} + \sum_{\{i \in \mathcal{R}[j,j+1], s_i > j+1\}} q_i x_{ik-1v} \leq Q \tag{4.18}$$

$$\sum_{v \in \{1,\ldots,V\}} \sum_{k \in \mathcal{K}_i} x_{ikv} = 1 \tag{4.19}$$

$$x_{ikv} \in \{0,1\} \tag{4.20}$$

with $1 \leq i \leq n, 0 \leq j \leq m-1, 1 \leq k \leq K, 1 \leq v \leq V$.

By summing capacity constraints on the m stations and V vehicles, we obtain surrogate capacity constraints:

$$\sum_{j=0}^{m-1} \sum_{v=1}^{V} \left(\sum_{\{i \in \mathcal{R}[j,j+1], s_i \leq j\}} q_i x_{ikv} + \sum_{\{i \in \mathcal{R}[j,j+1], s_i > j+1\}} q_i x_{ik-1v} \right) \leq mVQ \tag{4.21}$$

with $(1 \leq k \leq K)$.

The left-hand side can be rewritten:

$$\sum_{j=0}^{m-1} \sum_{\{i \in \mathcal{R}[j,j+1], s_i \leq j\}} q_i \left(\sum_{v=1}^{V} x_{ikv} \right) + \sum_{j=0}^{m-1} \sum_{\{i \in \mathcal{R}[j,j+1], s_i > j+1\}} q_i \left(\sum_{v=1}^{V} x_{ik-1v} \right) \tag{4.22}$$

that is,

$$\sum_{i \in \mathcal{R}} \sum_{\{s_i \leq j \leq m-1:\ i \in \mathcal{R}[j,j+1]\}} q_i \left(\sum_{v=1}^{V} x_{ikv} \right) + \sum_{i \in \mathcal{R}} \sum_{\{0 \leq j \leq s_i - 1:\ i \in \mathcal{R}[j,j+1]\}} q_i \left(\sum_{v=1}^{V} x_{ik-1v} \right) \tag{4.23}$$

Given a request i, the first part counts $q_i \left(\sum_{v=1}^{V} x_{ikv}\right)$ as many times as the number of interval covered by the request in the ring after s_i. The second part counts $q_i \left(\sum_{v=1}^{V} x_{ik-1v}\right)$ as many times as the number of intervals covered by the request in the ring before station s_i (between 0 and t_i). Equivalently, it can thus be rewritten:

$$\sum_{i \in \mathcal{R} \setminus \mathcal{R}^0} (t_i - s_i) q_i \left(\sum_{v=1}^{V} x_{ikv}\right) + \sum_{i \in \mathcal{R}^0} (m - s_i) q_i \left(\sum_{v=1}^{V} x_{ikv}\right) + \sum_{i \in \mathcal{R}^0} t_i q_i \left(\sum_{v=1}^{V} x_{ik-1v}\right) \tag{4.24}$$

where $\mathcal{R}^0$ is subset of requests that overlap station 0 (*i.e.*, $t_i < s_i$). In (4.23), the first two terms come from the first term of (4.22), the last term comes from the second term of (4.22).

Finally, thanks to (4.19), new binary variables $x_{ik} = \sum_{v=1}^{V} x_{ikv}$ can be introduced and the model becomes:

$$\min \sum_{i \in \mathcal{R}} \sum_{k \in \mathcal{K}_i} c_{ik} x_{ik} \tag{4.25}$$

s.t.:

$$\sum_{i \in \mathcal{R} \setminus \mathcal{R}^0} (t_i - s_i) q_i x_{ik} + \sum_{i \in \mathcal{R}^0} (m - s_i) q_i x_{ik} + \sum_{i \in \mathcal{R}^0} t_i q_i x_{ik-1} \leq mVQ \tag{4.26}$$

$$\sum_{k \in \mathcal{K}_i} x_{ik} = 1 \tag{4.27}$$

$$x_{ik} \in \{0, 1\} \tag{4.28}$$

with $1 \leq i \leq n, 1 \leq k \leq K$.

Results for this relaxation are not reported as it did not improve over formulation (4.6)-(4.9).

4.9 Conclusions

In this chapter new variants of the PDP-R were studied. We focused on problems in which vehicles travel in a single direction and the objective is to minimize the sum of completion times. All these variants were proven NP-hard.

An ILP formulation was proposed and computational tests were performed to evaluate its performance. Results show that the formulation is extremely effective, being able to solve to optimality almost all instances, even the largest ones with 160 requests, in a short computing time. On the most difficult variants of the problem, the formulation is always capable of providing a feasible solution when it exists and largely

outperforms a greedy algorithm mimicking common practice. Several techniques were explored to tackle the hardest instances, with no success at that point. A perspective of this work is to close these instances.

Chapter 5

The bi-directional PDP-R

Pickup and Delivery problems on rings (PDP-R) are a class of problems defined on cycle graphs. This chapter is devoted to the variants in which the vehicles are allowed to travel along both directions on the ring and the objective is to minimize the time at which the last vehicle comes back to the depot with all requests served. The simplest variant of this class of problems is polynomial-time solvable and is known in the literature as the Stacker Crane Problem on circles. In this chapter, we focus on this variant. Several algorithms have been proposed in the literature. We propose a new, easier, algorithm. Other variants are left for future works.

5.1 Introduction

Pickup and Delivery Problems on Rings (PDP-R) are defined on circular networks connecting a set of stations. A set of requests needs to be served, where each request consists of transporting a given quantity from a pickup station to a delivery station. A fleet of vehicles is available to perform the service and each vehicle route has to start and end at station 0, the depot. The goal is to determine the best transportation plan according to a given objective function.

PDP-R have been introduced in Trotta et al., 2022, where the authors propose a classification of the problem variants based on three fields $\alpha|\beta|\gamma$ where:

- α contains information about the number of vehicles and vehicle capacity;
- β specifies whether vehicles travel in one or both directions, whether release and due dates are defined and whether request have unitary demands;
- γ corresponds to the objective function.

In Trotta et al., 2022 applications of PDP-R are discussed: they are mainly linked to the use of autonomous vehicles for transportation services in concentrated areas (campus, industrial site, hospital...), for both people and freights. In addition, PDP-R also arise in the field of industrial automation. The reader is referred to Trotta et al., 2022 for a detailed overview of problem applications and related literature.

In Trotta et al., 2022, the authors introduce the PDP-R, propose a classification scheme and study the variant in which vehicles travel on a single direction and the objective function is the minimization of the time at which the last vehicle returns to the depot. They also show that the simplest problem in this class, *i.e.*, the problem with a single vehicle of unitary capacity and with no release and due date, is polynomially solvable while all others are NP-hard. In addition, they present a mathematical formulation for all variants which is proved to be effective through exhaustive computational tests.

In Trotta et al., 2021, the authors study PDP-R with cumulative cost, *i.e.* problems where the objective is the minimization of the sum of completion times of requests.

Contrary to what is done in Trotta et al., 2022 and Trotta et al., 2021, this chapter focuses on the variants where the vehicles travel in both directions along the ring and the objective function is the minimization of the closing time, i.e. the time at which the last vehicle comes back to the depot. Among all these variants, the problem with one vehicle of capacity one, no release dates, no due dates, is known in the literature as the Stacker Crane Problem. In the classification scheme proposed in Trotta et al., 2022, this problem is denoted by $1, 1|bd, u|CLT$. It has been proven to be polynomially solvable. However, the algorithms are relatively difficult to apprehend. In this chapter we focus on this problem and propose a new polynomial-time algorithm. The study of other variants is left for future works.

5.2 Literature review

This section reviews related contributions in the literature. The literature being quite limited, the section considers all variants of the class, i.e., Pickup and Delivery Problems on circular graphs in a bi-directional setting.

Guan, 1998 studies the multiple capacity non-preemptive vehicle routing problem on cycles. The author proves that the problem is NP-complete. Gendreau et al., 1999 and Tzoreff et al., 2002 study related freight transportation problems. The following is a list of similarities and differences between these problems and variants in the class of problems introduced in this chapter:

- Gendreau et al., 1999: the closest problem in our classification is $1, Q|bd|CLT$

where stations represent customers. However, there are some important differences:

  - in any feasible solution the vehicle leaves the depot with a load equal to the sum of delivery demands and gets back to the depot with a load equal to the sum of pickup demands;
  - if a customer requires both pickup and delivery, the two operations must be serviced at the same time;
  - the sum of pickup demands, as well as the sum of delivery demands, is smaller than Q.

  The consequence is that any feasible solution has optimal value $\leq 2L$ (with L length of the ring).

- Tzoreff et al., 2002: the authors study the same problem studied in Gendreau et al., 1999 but with possibly two depots on some special graphs, including cycle graphs. They also make similar hypothesis. These two papers study problems that can be transformed into a single instance of $1, Q|bd|CLT$.

A number of related problems arise in the field of industrial automation. In Atallah and Kosaraju, 1988 the authors were probably the first to study the problem of efficiently rearranging parts in the plane with a centrally placed gripper that can rotate. This problem is known as the Stacker Crane Problem (SCP). In this problem, a robot arm has the task of rearranging m objects among n stations positioned on a circular track. Each object is initially located at one of these stations and needs to be moved to another station. The robot arm consists of a single link that rotates around a fixed pivot. At the end of this link lies a gripper that is capable of grasping only one object at a time. Many objects can be located at a single station and can be moved to the same station. The gripper must terminate at the station where it started. It is assumed that the gripper moves only along the circumference of this circular track and that each station is the source or the destination of at least one object (*i.e.* $n \leq m$). The problem asks then to minimize the total length of the circular arcs traversed by the gripper. In their paper the authors show that an optimal transportation can be calculated in $O(m + nlogn)$ time.

In Frederickson, 1993 an $O(m+M(n,q))$ algorithm has been proposed for the same problem, where $q \leq \min\{n, m\}$ is the number of strongly connected components once additional augmenting edges are added to the input graph to make it semi-Eulerian and $M(a, b)$ is the time to solve a Minimum Spanning Tree problem on a graph with a edges and b vertices. Currently, the fastest known algorithm to solve the MST problem has time complexity $O(a\alpha(a, b))$, where α is the classical inverse Ackermann function (Chazelle, 2000).

In Anily and Pfeffer, 2013 a similar problem has been studied, the Uncapacitated Swapping Problem and on a circle, where the objective is to rearrange objects of different types on a circular graph using an uncapacitated vehicle. It can be seen as a generalization of the SCP. In their paper the authors proposed a polynomial time algorithm for both cases of a line and a circle.

Both algorithms proposed as solutions to the SCP on the circle are not easy to understand. This motivated us to propose an alternative algorithm that is easier to analyze and implement.

5.3 A simple algorithm for the Stacker Crane Problem on a circle

We start by defining the notation. We consider a bi-directional circle Γ, whose nodes are indexed by integers $0, 1, \ldots, n-1$, and a set $\mathcal{R}$ of m requests. Every arc a in the circle, between two consecutive vertices has a length c_a. Every request $r \in \mathcal{R}$ is defined by an origin $o(r)$, a destination $d(r)$ and a unit load. We call node 0 the depot. We also introduce node n as an alias of node 0. We impose that the vehicle starts from the depot and comes back to the depot after having served all the requests.

Without loss of generality, we assume that, except the depot, each node of Γ is the origin or the destination of at least 1 request. Indeed, useless stations can be easily eliminated with a pre-processing step that takes $O(m+n)$ time. This implies $n \leq 2m+1$.

We introduce the directed graph $G = (X, A)$, where $X = \{0, \ldots, n-1\}$ and $A = \{(o(r), d(r)) : r \in \mathcal{R}\}$. We note $\delta^+(x)$ the outdegree of vertex x in G, $\delta^-(x)$ its indegree and $\delta(x) = \delta^+(x) - \delta^-(x)$. We note c_a the length of arcs in A. Note that a request $o(r), d(r)$ can be executed clockwise or counter-clockwise on the circle: c_a is set to the minimal value between the two lengths.

Our algorithm is based on the same idea as the algorithm presented in Atallah and Kosaraju, 1988. The problem is modeled as a graph augmentation problem, *i.e.* the augmentation of graph G to an Eulerian graph such that the total length of the added arcs is minimum. To make the graph Eulerian, the new arcs are taken from the arc set $A_\Gamma = \{(x, x+1) : 0 \leq x \leq n-1\} \cup \{(x, x-1) : 1 \leq x \leq n\}$, *i.e.*, the arcs of the circle (or Γ-arcs). The vehicle can carry only one object at a time. In addition, in each optimal solution, as soon as the vehicle loads an object at the source station, it immediately drives to the destination station. Therefore, adding one of these Γ-arcs indicates that an empty move will be needed on the circle. Note that the same arc might be added several times and that the new graph is actually a multigraph, but for short we call it a graph. Any Eulerian cycle of the resulting Eulerian graph then

gives a solution. This solution is optimal in case the total length of the new arcs is minimized.

We call $\bar{G}$ the Eulerian graph constructed with our algorithm. To be Eulerian, $\bar{G}$ needs to be semi-Eulerian and strongly-connected. Semi-Eulerian means that the outdegree and the indegree are equal for all vertices. The transformation from G to $\bar{G}$ can be seen as the computation of an integral non negative vector U, indexed on the arcs of A_Γ, that tells the number of times each arc in A_Γ is added. U needs to be defined in such a way that:

1. $\bar{G}$ is semi-Eulerian: $\delta(x) + U_{(x,x-1)} + U_{(x,x+1)} - U_{(x-1,x)} - U_{(x+1,x)} = 0$ for $0 \leq x \leq n-1$, where $U_{(0,-1)}$ and $U_{(-1,0)}$ have to be read as $U_{(0,n-1)}$ and $U_{(n-1,0)}$;
2. $\bar{G}$ is strongly-connected;
3. $\sum_{a \in A_\Gamma} c_a U_a$ is minimised.

In Atallah and Kosaraju, 1988 vector U is called *optimal transportation*. We will now see how to compute a vector U that satisfies properties 1-3.

The basic idea of the algorithm is to transform the SCP on the circle into the SCP on the line. An algorithm on the line is summarily described in Atallah and Kosaraju, 1988 on pages 5-7, but neither a detailed description nor a proof of correctness are given. Therefore, we will first see how the SCP can be solved on the line (Section 5.3.1), and then explain how this algorithm can be adapted to the SCP on the circle (Section 5.3.2).

5.3.1 SCP on the line

The SCP on the line corresponds to the SCP on the circle described above except that arcs $(n-1, 0)$ and $(0, n-1)$ cannot be used. It also amounts to set $U_{(n-1,0)}$ and $U_{(0,n-1)}$ to 0. Incidentally, it also changes the cost of requests, because arcs $(n-1, 0)$ and $(0, n-1)$ can no longer be used.

To solve the SCP on the line, we proceed in two steps. We first transform G into a semi-Eulerian graph G^{SE}. Then, if G^{SE} has several connected components, $\bar{G}$ is obtained by connecting these components. In this section, we describe the algorithm but do not prove that it is correct nor evaluate its complexity. The proof and the complexity analysis are left to Sections 5.4 and 5.5, respectively.

We denote U^{SE} the vector describing the Γ-arcs added to transform G into G^{SE}. The key point of the algorithm is that values U^{SE}_a can be computed for every pair of arcs $\{(x, x+1), (x+1, x)\}$ in the increasing order of x $(0 \leq x \leq n-2)$. That is, the line graph can be balanced by iterating on the nodes in the increasing order of x. This is illustrated with Algorithm 3.

Algorithm 3 BALANCE_LINE

Input: G

1: $U^{SE}_{(0,1)} \leftarrow \max(0, -\delta(0))$
2: $U^{SE}_{(1,0)} \leftarrow \max(0, \delta(0))$
3: **for** $x = 1, \ldots, n-2$ **do**
4: $\quad U^{SE}_{(x,x+1)} \leftarrow \max(0, -\delta(x) - U^{SE}_{(x,x-1)} + U^{SE}_{(x-1,x)})$
5: $\quad U^{SE}_{(x+1,x)} \leftarrow \max(0, \delta(x) + U^{SE}_{(x,x-1)} - U^{SE}_{(x-1,x)})$
6: **end for**

Output: G^{SE}

The algorithm first initializes $U^{SE}_{(0,1)}$ and $U^{SE}_{(1,0)}$. Indeed, if $\delta(0) > 0$, the only arc available to recover a balanced degree at 0 is arc $(1, 0)$ and we add $\delta(0)$ copies of this arc. Conversely, if $\delta(0) < 0$ we add $-\delta(0)$ copies of arc $(0, 1)$. Once values $U^{SE}_{(0,1)}$ and $U^{SE}_{(1,0)}$ have been fixed, the same reasoning applies to vertex 1, and so on up to vertex $n-2$. The algorithm thus progressively computes values $U^{SE}_{(x,x+1)}$ and $U^{SE}_{(x+1,x)}$, for increasing values of x.

The next step is to merge the strongly connected components (s.c.c.) of graph G^{SE}. This is done with Algorithm 4 and with an ordered list $\mathcal{L}$ that has to be precomputed: $\mathcal{L}$ is the list of all pairs $(x, x+1)$, with $0 \leq x \leq n-2$ sorted in the increasing order of values $c_{(x,x+1)} + c_{(x+1,x)}$.

Algorithm 4 CONNECT

Input: G^{SE}

1: compute set $\mathcal{C}$ of s.c.c.'s in graph G^{SE}; denote $\mathcal{C}(x)$ the s.c.c. for $x \in X$
2: $U \leftarrow U^{SE}$
3: **for all** $(x, x+1)$ taken in the order of $\mathcal{L}$ **do**
4: $\quad$ **if** $\mathcal{C}(x) \neq \mathcal{C}(x+1)$ **then**
5: $\quad\quad U_{(x,x+1)} \leftarrow 1$
6: $\quad\quad U_{(x+1,x)} \leftarrow 1$
7: $\quad\quad$ **for all** u such that $\mathcal{C}(u) = C(x+1)$ **do**
8: $\quad\quad\quad \mathcal{C}(u) \leftarrow \mathcal{C}(x)$
9: $\quad\quad$ **end for**
10: $\quad$ **end if**
11: **end for**

Output: $\bar{G}$

The algorithm initially calculates the set of strongly connected components, for example applying the algorithm described in Cormen et al., 2009, p. 617. It then iterates over the arcs $(x, x+1)$ in $\mathcal{L}$, according to its order. At any iteration, if nodes

x and $x+1$ belong to different connected components, a pair of arcs $(x, x+1)$ and $(x+1, x)$ is added and the corresponding s.c.c.'s are merged. Note that lines 5 and 6 actually correspond to increasing by one values $U_{(x,x+1)}$ and $U_{(x+1,x)}$, but the fact that x and $x+1$ belong to different s.c.c.'s implies $U_{(x,x+1)} = U_{(x+1,x)} = 0$.

The following is a description of the general algorithm for the SCP on the line.

Algorithm 5 SCP-LINE

Input: request graph G with n nodes and m requests
1: **for all** $x = 0, \ldots, n-1$ **do**
2: compute $\delta(x)$
3: **end for**
4: compute ordered list $\mathcal{L}$
5: $G^{SE} \leftarrow$ BALANCE_LINE(G)
6: $\bar{G} \leftarrow$ CONNECT(G^{SE})
7: compute an Eulerian cycle $\bar{G}^E$ of $\bar{G}$
Output: $\bar{G}^E$

5.3.2 SCP on the circle

Dealing with the circle prevents from using the algorithm defined on the line for two reasons:

1. The initialization of Algorithm 3 is not possible because, in addition to arcs $(0,1)$ and $(1,0)$, arcs $(n-1,0)$ and $(0,n-1)$ can also help recovering a balanced degree at node 0. Deciding which arcs should be used is not straightforward.

2. The algorithm used to connect graph G^{SE} is not valid because it might be better to introduce a complete circuit, clockwise or counterclockwise, instead of introducing pairs of arcs of opposite direction as it is done in Algorithm 4.

To deal with the first difficulty, we arbitrarily set values to $U^{SE}_{(n-1,0)}$ and $U^{SE}_{(0,n-1)}$. Then, given these values, Algorithm 3 can be applied. The process is repeated for every possible pair of values that $U^{SE}_{(n-1,0)}$ and $U^{SE}_{(0,n-1)}$ can assume in any optimal solution. We call Δ this set of pairs, whose construction is explained later.

The second issue, about connectivity, does not raise a significant difficulty. Given a pair $(U^{SE}_{(n-1,0)}, U^{SE}_{(0,n-1)}) \in \Delta$, the solution obtained when a clockwise (resp., counterclockwise) circuit is added, is actually captured in pair $(U^{SE}_{(n-1,0)} + 1, U^{SE}_{(0,n-1)})$ (resp., in pair $(U^{SE}_{(n-1,0)}, U^{SE}_{(0,n-1)} + 1)$). So, it is important to consider this possibility in the definition of set Δ but then, the algorithm connecting the graph does not have to

be modified: the possibility that the graph is connected with a complete circuit will appear while making the graph semi-Eulerian.

These modifications translate into Algorithms 6 and 7.

Algorithm 6 BALANCE-CIRCLE

Input: G, $U^{SE}_{(n-1,0)}$, $U^{SE}_{(0,n-1)}$
1: **for** $x = 0, \ldots, n-2$ **do**
2: $\quad U^{SE}_{(x,x+1)} \leftarrow \max(0, -\delta(x) - U^{SE}_{(x,x-1)} + U^{SE}_{(x-1,x)})$
3: $\quad U^{SE}_{(x+1,x)} \leftarrow \max(0, \delta(x) + U^{SE}_{(x,x-1)} - U^{SE}_{(x-1,x)})$
4: **end for**
Output: G^{SE}

Algorithm 7 SCP-CIRCLE

Input: request graph G with n nodes and m requests
1: $c_{best} \leftarrow +\infty$
2: **for all** $x = 0, \ldots, n-1$ **do**
3: $\quad$ compute $\delta(x)$
4: **end for**
5: compute ordered list $\mathcal{L}$
6: compute set Δ
7: **for all** pairs of values $(U^{SE}_{(n-1,0)}, U^{SE}_{(0,n-1)})$ in Δ **do**
8: $\quad G^{SE} \leftarrow$ BALANCE_CIRCLE$(G, U^{SE}_{(n-1,0)}, U^{SE}_{(0,n-1)})$
9: $\quad \bar{G} \leftarrow$ CONNECT(G^{SE})
10: $\quad$ update c_{best} and $\bar{G}_{best}$ if the cost of an Eulerian cycle in $\bar{G}$ is better
11: **end for**
12: compute an Eulerian cycle $\bar{G}^E$ of $\bar{G}_{best}$
Output: $\bar{G}^E$

Algorithm 6 only differs from Algorithm 3 in the initialization of values $U^{SE}_{(0,1)}$ and $U^{SE}_{(1,0)}$ that take into account values $(U^{SE}_{(n-1,0)}, U^{SE}_{(0,n-1)})$. Algorithm 7 first computes values $\delta(x)$, list $\mathcal{L}$ and set Δ. Then for every pair $(U^{SE}_{(n-1,0)}, U^{SE}_{(0,n-1)})$ in Δ, it makes the graph semi-Eulerian (thanks to Algorithm 6), connects it (thanks to Algorithm 4) and obtains an Eulerian graph $\bar{G}$. It saves the graph with the best circuit and compute an Eulerian cycle in this graph when all pairs in Δ have been tried.

Set Δ is defined with the following pairs $(U^{SE}_{(n-1,0)}, U^{SE}_{(0,n-1)})$:

1. $(p, 0)$ with $1 \leq p \leq m+1$;
2. $(0, q)$ with $1 \leq q \leq m+1$;

3. $(1, 1)$;
4. $(0, 0)$.

We will see in the next section why it guarantees finding the optimal solution. Especially, we will see that larger values for p or q imply having "parallel" circuits of Γ-arcs, which is clearly not optimal. Also, having values larger than 1 for both $U^{SE}_{(n-1,0)}$ and $U^{SE}_{(0,n-1)}$ imply having a short circuit $(n-1, 0, n-1)$ than can also be removed without losing the connectivity.

5.4 Correctness of Algorithms 5 and 7.

We now show that Algorithms 5 and 7 are correct, that is, they calculate an optimal solution for the SCP on the line and on the circle, respectively.

Remark 1. *Note that the only arcs that need to be added in Algorithm 3 are the Γ-arcs of the form $(x, x+1)$ or $(x+1, x)$. Indeed, an arc that covers i intervals can always be broken into i Γ-arcs without increasing the total arc length and without losing the balance in any node.*

Lemma 9. *Algorithm 3 makes graph G semi-Eulerian by adding a set of arcs that necessarily appear in any optimal solution.*

Proof. The proof is by induction on the number of nodes.

- Base case: $x = 0$. If $\delta(0) > 0$, the only arc available to balance node 0 is arc $(1, 0)$ and we add $\delta(0)$ copies of this arc. If instead $\delta(0) < 0$, the only arc that can be used to balance node 0 is arc $(0, 1)$ we add $-\delta(0)$ copies of arc $(0, 1)$. In both cases, arcs $(0, 1)$ and $(1, 0)$ are the only available (see Remark 1) and it is not possible to recover a balanced degree at node 0 adding less arcs.
- Induction step: suppose the algorithm is at an iteration such that $x = n'$ and suppose the smallest number of mandatory arcs have been added up to $x = n' - 1$. The algorithm has to decide how many arcs of the type $(n', n'+1)$ or $(n'+1, n')$ to add. If $\delta(n') > 0$, the only arc available to balance node n' is arc $(n', n'+1)$ (by hypothesis the smallest number of arcs $(n'-1, n')$ and $(n', n'-1)$ have already been added) and $\delta(n')$ copies of this arc are added. If instead $\delta(n') < 0$, the only arc that can be used to balance node n' is arc $(n'+1, n')$ (for the same reason) and $-\delta(n')$ copies of arc $(n'+1, n')$ are added. In both cases, arcs $(n', n'+1)$ and $(n'+1, n')$ are the only available (for Remark 1) and the smallest number of them is added ($|\delta(n')|$).

□

Lemma 10. *Algorithm 4 makes graph G^{SE} Eulerian by connecting the s.c.c. with arcs of minimum total length.*

Proof. It suffices to note that the problem of connecting strongly connected components with arcs of minimum total cost is equivalent to computing a Minimum Spanning Tree (MST) on the undirected graph in which the nodes are the s.c.c.'s and there is an edge (x, y) between a pair of s.c.c.'s i and j if and only if $x \in i$ and $y \in j$ and x is adjacent to y (*i.e.* $|x - y| = 1$). The length of the edge (x, y) corresponds to the sum of distances between stations x and y, that is $c_{(x,x+1)} + c_{(x+1,x)}$ (note that also the arithmetic average could be taken as a measure). If there are multiple edges connecting two s.c.c. i and j, then the edge (x, y) of minimum length is chosen. Then, algorithm 4 is equivalent to applying Kruskal's algorithm to calculate an MST.

□

Lemma 9 proves that Algorithm 5 transforms a line graph into a semi-Eulerian graph G^{SE} by adding the smallest number of mandatory arcs. Lemma 10 proves that the same algorithm transforms G^{SE} into an Eulerian graph by adding arcs of minimum total length. Then, it computes an Eulerian cycle on this graph, that is, an optimal solution. As a consequence, Algorithm 5 is correct. We now prove that algorithm 7 is also correct. We first need two simple lemmas.

Lemma 11. *For all $x \in \{0, \dots, n-1\}$, $\sum_{i=0}^{x} \delta(i) \leq m$.*

Proof. $\sum_{i=0}^{x} \delta(i) \leq \sum_{i=0}^{x} \delta^+(i) = \sum_{i=0}^{x} |\{r \in \mathcal{R} : o(r) = i\}| \leq |R| = m.$ □

Lemma 12. *Let $x \in \{0, \dots, n-1\}$. In any optimal solution, $U_{(x,x+1)} \geq 2$ implies $U_{(x+1,x)} = 0$. Similarly, $U_{(x+1,x)} \geq 2$ implies $U_{(x,x+1)} = 0$*

Proof. We prove by contradiction the first part of the Lemma, the proof of the second part is similar. Assume an optimal solution with $U_{(x,x+1)} \geq 2$ and $U_{(x+1,x)} \geq 1$. The solution contains a circuit $(x, x+1, x)$ composed entirely of Γ-arcs that we can remove to obtain a Eulerian cycle of smallest total length, without losing neither the strong connectivity nor the semi-Eulerian property. This contradicts the hypothesis that vector U is optimal. □

Lemma 13. *Any optimal solution to the SCP on the circle satisfies one (and only one) among configurations 1-4.*

Proof. Consider an optimal vector U, *i.e.* a vector associated with an optimal solution of the SCP on the circle. We first observe that it is impossible that $U_{(0,n-1)} \geq 1$ and $U_{(n-1,0)} \geq 2$ (or vice-versa) thanks to Lemma 12. This proves that either one among

$U_{(n-1,0)}$ and $U_{(0,n-1)}$ is equal to 0 (*i.e.* configurations 1-3) or that they are both equal to 1 (configuration 4). It thus remain to prove:

1. $U_{(0,n-1)} = 0 \implies U_{(n-1,0)} \leq m+1$
2. $U_{(n-1,0)} = 0 \implies U_{(0,n-1)} \leq m+1$

We only prove 1, the proof of 2 is identical. The proof is by contradiction. Suppose $U_{(0,n-1)} = 0$ and $U_{(n-1,0)} \geq m+2$. We prove that the solution cannot be optimal. We first prove by induction that, for all $x \in \{0,\ldots,n-1\}$, $U_{(x,x+1)} \geq m+2-\sum_{i=0}^{x}\delta(i)$.

- Base case ($x = 0$): We know that in any optimal solution, $U_{(1,0)} + U_{(n-1,0)} + \delta^-(0) = \delta^+(0) + U_{(0,1)} + U_{(0,n-1)}$. By hypothesis, $U_{(0,n-1)} = 0$ and we know $\delta^+(0) - \delta^-(0) = \delta(0)$. So, $U_{(n-1,0)} = \delta(0) + U_{(0,1)} - U_{(1,0)}$. From the hypothesis, we have $U_{(n-1,0)} \geq m+2$, which implies $U_{(0,1)} - U_{(1,0)} \geq m+2-\delta(0)$ and, then, $U_{(0,1)} \geq m+2-\delta(0)$.
- Induction step: consider $x \in \{1,\ldots,n-1\}$ and suppose that $U_{(x-1,x)} \geq m+2-\sum_{i=0}^{x-1}\delta(i)$. Note that using lemmas 11 and 12, it also shows that $U_{(x-1,x)} \geq 2$ and that $U_{(x,x-1)} = 0$. We know that $U_{(x+1,x)} + U_{(x-1,x)} = \delta(x) + U_{(x,x+1)} + U_{(x,x-1)}$. So, $U_{(x,x+1)} \geq m+2-\sum_{i=0}^{x-1}\delta(i) + U_{(x+1,x)} - \delta(x) - U_{(x,x-1)}$. It follows $U_{(x,x+1)} \geq m+2-\sum_{i=0}^{x}\delta(i)$.

The induction proves that $U_{(x,x+1)} \geq 2$ for $x = 0,\ldots,n-1$. It implies the existence of two "parallel" clockwise circuits made only with Γ-arcs, one of which we could remove without losing neither the strong connectivity nor the semi-Eulerian property, thus obtaining an Eulerian cycle of smaller total length. This contradicts the fact that the solution is optimal. □

A consequence of lemma 13 is that an optimal solution to the SCP on the circle can be found by solving $|\Delta|$ instances of the SCP on the line, where each instance is obtained for different values of vector U in set Δ. This proves Algorithm 7 is correct.

5.5 Complexity analysis

We now analyze the time complexity of Algorithms 5 and 7.

Algorithm 5

The complexity is as follows:

- Values $\delta(x)$ in lines 1-3 can be computed in $O(n+m)$ by initializing $\delta(x)$ to 0 for all x and, then, scanning all requests, with $\delta(o(r))$ increased and $\delta(d(r))$ decreased when request r is scanned;
- Ordered list $\mathcal{L}$, of size n, is computed in $O(n \log n)$;
- The call to Algorithm 3 is $O(n)$;
- The call to Algorithm 4 is $O(m + n\alpha(n))$, where α is the inverse Ackermann function:
  - In line 1 of Algorithm 4, the set of s.c.c.'s can be computed in linear time on the size of graph G^{SE} in which parallel Γ-arcs are merged, that is, in $O(m+n)$, see Cormen et al., 2009, p. 617;
  - If disjoint-sets (or union-find) data structures are used, lines 3-11 of Algorithm 4 can be executed in $O(n\alpha(n))$. In fact, a sequence of n merge operations has time complexity of $O(n\alpha(n))$, see Cormen et al., 2009, pp. 561-572.
- Finding an Eulerian cycle (line 7) can be done in $O(m+t)$, where t is the number of Γ-arcs in graph $\bar{G}$.

The overall complexity of Algorithm 5 is then:

$$O(n+m) + O(n \log n) + O(n) + O(m + n\alpha(n)) + O(m+t)$$

that is,

$$O(n \log n + n\alpha(n) + m + t)$$

Algorithm 7

The complexity is as follows:

- Values $\delta(x)$ in lines 2-4 can be computed in $O(n+m)$;
- Ordered list $\mathcal{L}$, of size n, is computed in $O(n \log n)$;
- Set Δ is computed in $O(m)$ and has a size $O(m)$; so, lines 8 to 10 are repeated $O(m)$ times:
  - The call to Algorithm 6 is $O(n)$;
  - The call to Algorithm 4 is still $O(m + n\alpha(n))$;
  - Computing the cost of an Eulerian cycle and updating the graph is $O(n)$ with the graph coded with vector U;

- Computing an Eulerian cycle is $O(m + t)$,using notation t introduced before.

The overall complexity of Algorithm 7 is therefore:

$$O(n + m) + O(n \log n) + O(m) + O(m(n + m + n\alpha(n) + n)) + O(m + t)$$

that is,

$$O(n \log n + mn\alpha(n) + m^2 + t)$$

Finally, the number of arcs added between two consecutive nodes of the cycle is $O(m)$, so $O(t) = O(mn)$ and the overall complexity of Algorithm 7 is:

$$O(n \log n + mn\alpha(n) + m^2)$$

It is less efficient than the algorithms proposed in Atallah and Kosaraju, 1988 and Frederickson, 1993. However, it has the advantage of being easy to understand and implement.

5.6 Conclusion

In this chapter a new class of Pickup and Delivery Problems on rings was introduced and studied. In this class of problems, vehicles have the ability to move on the ring following both directions of rotation. The simplest variant in this class of problems, is known in the literature as the Stacker Crane Problem (SCP). Two algorithms for SCP have already been proposed in the literature, but neither of them is easy to understand and implement. A simple algorithm for the SCP was proposed. The other problems in this class can be seen as generalizations of SCP.

Our work opens the way for multiple research possibilities. A first perspective could be to to improve the complexity of Algorithm 7. Another perspective is to study the complexity of the variants not studied in this chapter.

Chapter 6

The Electric Vehicle Pickup and Delivery Problem with Energy Management

In this chapter a problem in which a set of capacitated Battery Electric Vehicles (BEVs) carry out pickup and delivery operations with time windows constraints is studied. The energy needed to recharge the batteries of these vehicles is produced in a Battery Swapping Station (BSS) that is also the depot of the vehicles. Additional batteries are available at the depot, where vehicles can go and swap their batteries. Pickup and delivery operations must be planned over a time horizon divided into periods. In each period it must be decided how much energy to give to the batteries that are at the production unit. Also, if the energy produced is in excess of that required by the batteries, this excess can be sold to the general network at a profit. If the energy required by the batteries is greater than the energy produced, an unlimited amount of energy can be bought from the general network. The objective of the problem is to plan vehicle routes to meet all pickup and delivery demands while maximizing the net profit that is made from the energy sold and bought over the time horizon. An MILP formulation of the problem is proposed and a matheuristic approach is developed. The matheuristic approach consists of three steps: in the first one a subset of feasible trips is generated by using a Randomized Construction Heuristic, in the second step the formulation is solved over this set of trips, and in the third one a repair procedure is performed on the obtained solution, in order to avoid more than one trip visiting the same node. Computational tests on modified Li and Lim's benchmark instances for the PDPTW (H. Li & Lim, 2003) are performed and the impact of the parameters on the hardness of these instances is studied.

6.1 Introduction

The use of electric vehicles has heavily increased in the last years. In June 2022, the European Parliament approved a ban on the sale of all vehicles with Internal Combustion Engines (ICEs) from 2035 (ENVI, 2022). In reality, the measure approved by the Parliament aims to sell only CO2 emissions-free vehicles from 2035, but the lack of widely available alternatives means that the market for new cars will be dominated by Battery Electric Vehicles (BEVs) (McKinsey, 2022).

The European Automobile Manufacturers Association (ACEA), described the plan as ambitious and called for drastic action on building the charging infrastructure. According to ACEA, the key to reaching CO2 targets is not in the industry's hands alone – others need to play their part too. In particular, they emphasise the importance of the availability of the key raw materials for e-mobility and a truly EU-wide network of charging and refuelling infrastructure (ACEA, 2022).

The expansion of the electric vehicle charging network presents some critical issues. According to the McKinsey Center for Future Mobility, Europe will have to build an estimated 24 new battery giga-factories to supply EV battery demand. With more than 70 million EVs on the road by 2030, the industry will need to install a large number of public chargers and provide maintenance operations for them. Renewable electricity production needs to increase by 5% to meet EV charging demand (McKinsey, 2022).

Charging stations for EVS have two main disadvantages: long charging times and high power consumption at peak times. Rapid charging stations solve the problem of long charging times but have several battery degradation effects as consequences (Tomaszewska et al., 2019).

An innovative way is to refuel the energy source of EVs by mechanically swapping the batteries. These swapping stations are known as Battery Swapping Stations (BSS) (see figure 6.1) where the discharged batteries are swapped with fully charged batteries. With the use of robotic machinery, the whole battery swapping process can be carried out within a few minutes, directly comparable to the existing refuelling mechanism for conventional vehicles. Advantages of BSS include recharging the vehicle in a shorter time and charging during off-peak periods (Ahmad et al., 2020; Ding et al., 2022). BSS has also significant potential to work as a grid scale energy storage (Hosseini et al., 2018; Revankar & Kalkhambkar, 2021).

Figure 6.1: Blue Park Smart Technology's battery swap station in China

There are many obstacles to the practical implementation of battery swapping. Firstly, the initial cost to set up this battery swapping system is very high, involving expensive robotic machinery to swap the battery and a large number of costly batteries for necessary operation. Secondly, due to the need to store both discharged and fully charged batteries, the necessary space to build a battery swapping station is much larger than that for a charging station. Thirdly, the EV batteries need to be standardized in physical dimensions and electrical parameters before the possible implementation of automatic battery swapping (Chau, 2014).

A solution to some of the disadvantages of BSS are solar-powered BSSs. In this type of BSS, energy is produced on-site through the use of PhotoVoltaic (PV) panels. The use of PV panels allows the high initial investment cost of installing BSSs to be amortized over time. At the same time, this type of BSS offers the advantage of using solar energy, which is a renewable energy source.

Gogoro is a Taiwanese company that developed a battery-swapping refueling platform for urban electric two-wheel scooters, mopeds and motorcycles. In 2017 they launched their first solar-powered station in the Bali district of New Taipei City, Taiwan (Gogoro, 2017). This station is equipped with eight 2.3kw solar panels, generating up to 6.21 kwh per day, which is enough to charge up to ten batteries per day (see figure 6.2).

Figure 6.2: Gogoro's solar-powered battery-swapping stations in Taiwan (Gogoro, 2017)

The current chapter focuses on the optimization of on-demand transportation services in which battery electric vehicles are used. Specifically, in this chapter the aim is to study pickup and delivery problems where the energy used to recharge the batteries of electric vehicles is produced by a grid-connected photovoltaic power station. The power plant is energy-autonomous (most of the time), depending on the availability of the sun light. The vehicles swap their batteries at a BSS that is also the depot. Pickup and delivery operations are associated with release and due dates, which are respectively earliest time for pickup and the latest time for delivery. The problem studied is a multi-period problem where the time horizon is one day and each period corresponds to one hour. Each hour of the day is associated with different costs for buying and selling energy. The power plant produces energy through the use of photovoltaic panels. This energy can be either used to recharge the batteries of electric vehicles or sold to the electricity grid. In addition, should the electricity produced not be sufficient to recharge the batteries, additional energy can be purchased from the electricity grid. The objective of the problem is to serve all transport requests by maximizing the profit generated by the difference between the sale and purchase of electricity over all periods of the time horizon.

The rest of the chapter is organized as follows: in Section 6.2 the relevant literature is surveyed. In Section 6.3 a formal description of the problem and an MILP formulation are presented. In Section 6.4 a 3-phase solution method is presented. Section 6.5 contains the results of the computational tests. Finally, in Section 6.6, the conclusions and some perspectives of this work are presented.

6.2 Literature review

The literature on routing problems with electric vehicles (EVs) is first revised. Starting from 2011, several variants of the VRP have been studied, among which the most important is the Electric VRP (E-VRP) (see, for example, Lin et al., 2016). In this class of problems, the limitations of EVs are added to the classical constraints of the VRP, like the limited battery capacity or the need for recharging stops. In the last ten years a great number of papers have studied routing problems with electric vehicles (Barco et al., 2017; T. Chen et al., 2018; Koç et al., 2019; H. Yang et al., 2015). The reader is referred to Erdelić and Carić, 2019; Qin et al., 2021 for recent surveys on the E-VRP and Pelletier et al., 2016 for a survey on freight distribution problems with EVs.

EVs are generally divided into three categories: battery EVs (BEVs), hybrid electric vehicles (HEVs), and fuel cell electric vehicles (FCEVs) (Pelletier et al., 2016). With respect to Internal Combustion Engine Vehicles (ICEVs), some advantages are associated with BEVs: the ability to recuperate kinetic energy during the break (known as regenerative breaking), local-free emissions, high tank-to-wheel efficiency (approx. three times higher than ICEVs), very high acceleration. However, they have disadvantages as well: for example, shorter distance ranges and high battery recharging times (Helmers & Marx, 2012; Herrmann & Rothfuss, 2015). Anyway, it is realistic to consider battery-swapping scenarios in which vehicles can swap their batteries in a few minutes, even if battery-swapping stations have some disadvantages like high operational costs (Ulrich, 2021). A certain number of papers address routing problems in a battery-swapping scenario (Adler & Mirchandani, 2014; J. Chen et al., 2016; Jie et al., 2019; J. Li et al., 2020; Mao et al., 2020; Raeesi & Zografos, 2020; Sayarshad et al., 2020; Verma, 2018). Routing BEVs is more complex than routing ICEVs. The reason is that the energy consumption model of electric vehicles is much more complicated. For example, as demonstrated in Lin et al., 2016, the vehicle load has an effect on energy consumption. Many authors propose an energy consumption model suited for electric vehicles (Baek et al., 2019; Fiori & Marzano, 2018; Xiao et al., 2019).

As we are interested in Pickup and Delivery problems (PDP) with EVs, the most relevant literature in this field is briefly cited. Gonçalves et al., 2011 study pickup and delivery problems with a mixed fleet of BEVs and ICEVs. Grandinetti et al.,

2016 adds time windows to the PDP and study the electric Pickup and Delivery Problem with Time Windows (E-PDPTW). In Goeke, 2019 the authors propose a metaheuristic approach for the same problem. A similar problem is also studied in Ghobadi et al., 2021, in which the authors introduce multiple depots and fuzzy time windows. Pickup and Delivery problems with electric vehicles are also studied in Barco et al., 2017; Ghobadi et al., 2021; Jung and Jayakrishnan, 2012; Lu et al., 2018; Outalha, Abdelhak et al., 2021; Soysal et al., 2020; Wang and Cheu, 2013; Q.-q. Yang et al., 2018.

In the context of people transportation, PDP is usually referred to as the Dial-a-Ride problem (DARP). Since the DARP deals with people transportation, in this problem user-maximum ride times are incorporated into classic objective functions. The Green VRP has a corresponding version in the context of people transportation, namely the Green Dial-a-Ride problem (G-DARP) (Abedi et al., 2019; Atahran et al., 2014; Maggi & Faizrahnemoon, 2014; Masmoudi et al., 2019; Yu et al., 2019) in which the problem is referred to as green ride-sharing. In Bongiovanni et al., 2019 a variant of the DARP with electric autonomous vehicles is studied (e-ADARP). Dial-a-Ride problems with electric autonomous vehicles are also studied in Pimenta et al., 2017b, but the authors do not take into consideration battery recharging times. Other contributions can be found in Chabrol et al., 2008; Ma, 2021; Masmoudi et al., 2020; Perera et al., 2018. In Masmoudi et al., 2018 the authors address the DARP with electric vehicles and a battery swapping policy.

The second type of literature relevant for the current chapter concerns *integrated problems*, *i.e.* interdependent problems that have to be jointly optimized. Indeed, in the problem studied, one has to jointly optimize energy production and transportation operations.

A relevant integrated problem is the inventory routing problem. It is a distribution problem in which each customer maintains a local inventory of a product and consumes a certain amount of that product in each day. Every day a fleet of trucks is dispatched over a set of routes to resupply a subset of the customers. As first defined in Dror et al., 1985, the objective is to minimize the delivery costs, while ensuring that no customer runs out of the product at any time. The inventory routing problem integrates two classic problems in logistics, *i.e.* inventory management and transportation. For a survey on inventory routing problems, see Coelho et al., 2014. One generalization of the IRP is the IRP with Pickups and Deliveries (IRP-PD) (Archetti et al., 2018; Archetti et al., 2020; van Anholt et al., 2016). In this problem, a single commodity has to be picked up from several origins and distributed to several destinations. The commodity is made available at the supplier depot and at the pickup customers, and it is consumed by the delivery customers. The objective is to determine a collection and distribution plan minimizing the sum of routing and inventory costs, satisfying inventory and capacity constraints. In Iassinovskaia et al., 2017 a similar problem is studied, that the authors call the Pickup and Delivery

Inventory Routing Problem with Time Windows (PDIRPTW), in which a fleet of vehicles distributes two types of commodities from a supplier to a set of customers. The customers also specify time windows in which they are available.

Another relevant integrated problem is the Production Routing Problem. It consists in simultaneously optimizing production, distribution, inventory management. It can therefore be considered as a generalization of the IRP. The PRP has received a lot of attention in the research community (Absi et al., 2015; Absi et al., 2018; Adulyasak et al., 2014). In Golsefidi and Jokar, 2020, the authors study a variant of the PRP with simultaneous pickups and deliveries, where a supplier distributes some products to a set of customers while at the same time collecting the defective products. For a survey on the PRP, see Adulyasak et al., 2015.

In the context of the synchronization of energy production and vehicle routing, a relevant paper is Bendali, Fatiha et al., 2021. In their paper, the authors address the problem of the simultaneous management of a fleet of small electric vehicles provided with hydrogen power cells, which perform pickup and delivery operations inside a restricted area, and a micro-plant in charge of producing the hydrogen fuel which is going to be periodically loaded into the vehicles. The aim is to match the pickup and delivery activity of the vehicles with the hydrogen production/stock strategy of the micro-plant. The authors consider only one vehicle, which is required to perform tasks according to a pre-fixed order. This vehicle starts its route with some hydrogen fuel load, and its tank has a limited capacity. Therefore, it must periodically go back to the micro-plant in order to refuel. The micro-plant has a limited production/storage capacity, which depends on solar illumination. The goal is to simultaneously schedule both the refueling transactions of the vehicle and the production/storage activity of the micro-plant, while minimizing production economical cost and the duration of the vehicle tours.

There already exists a few papers on integrated problems that combine routing decisions for electric vehicles with other types of decisions: inventory planning and vehicle routing (Ni et al., 2021), location of charging stations and vehicle routing (Worley et al., 2012), battery swap location and vehicle routing (Hof et al., 2017; J. Yang & Sun, 2015).

In Widrick et al., 2018 an EV swap station allows EV owners to quickly exchange their depleted battery for a fully charged battery. The authors introduce the EV-Swap Station Management Problem (EV-SSMP), which models battery charging and discharging operations at an EV swap station facing nonstationary, stochastic demand for battery swaps, nonstationary prices for charging depleted batteries, and nonstationary prices for discharging fully charged batteries giving energy to the grid. The objective of the EV-SSMP is to determine the optimal policy for charging and discharging batteries that maximizes expected total profit over a fixed time horizon. In Mahoor et al., 2019 the authors propose a mathematical model for uncertainty-

constrained BSS optimal operation that not only covers the random customer demands of fully charged batteries, but also leverages the available batteries to reduce its operation cost through demand shifting and energy sellback. In Justin et al., 2020 the authors propose power optimized and power-investment optimized strategies for electric aircraft battery swaps and recharges. Several aspects are considered: electric energy expenditures, capital expenditures, and flight schedule integrity. In Kang et al., 2016 a novel centralized charging strategy of EVs under the battery swapping scenario is proposed by considering optimal charging priority and charging location (station or bus node in a power system) based on spot electric price. In Liu et al., 2015, a charging strategy considering the service availability and self-consumption of the PV energy is proposed to improve the operation performance of the PV-based BSS. The charging strategy consists in a battery-swapping service model and power distribution model. In Nurre et al., 2014 the authors consider a deterministic integer programming model for determining the optimal operations of multiple plug-in hybrid electric vehicle (PHEV) battery exchange stations over time. The decisions include the number of batteries to charge, discharge, and exchange at each point in time over a set time horizon. Discharging of batteries back to the power grid is allowed through vehicle-to-grid technology. In Cheng and Tao, 2018 a method to optimize micro energy network integrated with bus swapping station and distributed PV resources is proposed. The charging model is aimed at minimizing cost of electricity purchasing. Based on the electrification of a bus line in a city, the charging load characteristics and the former charging strategy are studied. The authors propose an optimization model for EV battery swapping station considering PV consumption bundling requirement. One day's bus line operation is simulated.

All of the cited articles either model the demand for battery swaps as random variables or use calculated demands based on already planned vehicle routes. In this chapter it is addressed the problem of optimizing battery charging operations and buying/selling the energy produced simultaneously with optimizing electric vehicle routes. The main contributions of this chapter can be summarized as follows:

- The problem of optimizing pickup and delivery operations and managing a PV-BSS where electric vehicles exchange their batteries is introduced. Management of the PV-BSS includes recharging the batteries at various periods of the time horizon and the eventual purchase or sale of the energy produced by the power plant.
- An MILP formulation for the problem is proposed, based on the concept of a trip: a trip is the route of a vehicle associated with a start period and an end period. Each trip starts from the depot, serves some transportation requests and returns to the depot. In the mathematical formulation, the routing problem is modeled as a trip choice problem.
- The problem is solved with a matheuristic consisting of 3 steps. In the first

step a constructive heuristic is used to generate a subset of feasible trips, in the second step the MILP is solved with CPLEX on the set of trips generated in the first step, and in the third step a solution repair procedure is applied in case some requests are contained in more than one trip.

- The performance of the formulation is evaluated through some preliminary computational tests conducted on the modified Li and Lim's benchmark instances for PDPTW. In these tests it is studied the impact that the number of trips have on the computation time of a feasible solution for the formulation.

6.3 Problem description and MILP modelling

The problem can be formally described as follows. We have a complete directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the set of all nodes and $\mathcal{A} = \{(i, j) : i, j \in \mathcal{N}, i \neq j\}$ is the set of arcs connecting each pair of nodes. The set of nodes $\mathcal{N} = \{0, 2n + 1\} \cup \mathcal{P} \cup \mathcal{D}$ consists of two copies of the depot/production plant $(0, 2n + 1)$, the set of pickup nodes $\mathcal{P} = \{1, \ldots, n\}$ and the set of delivery nodes $\mathcal{D} = \{n + 1, \ldots, 2n\}$. Every route begins and ends at the depot. There are n transportation requests that have to be served over a planning horizon $\mathcal{H}$ of H periods $\{1, \ldots, H\}$, each of duration τ. Periods can be interpreted as hours and the time horizon can be interpreted as a day.

An arc (i, j) in set $\mathcal{A}$ has an associated non-negative energy cost e_{ij} and a non-negative travel time τ_{ij}. We assume that a fleet $\mathcal{V}$ of $|\mathcal{V}| = V$ homogeneous BEVs is available at the depot, each of capacity Q. Vehicles can swap their batteries at the depot. We assume that vehicles move at constant speed on the network. K batteries are available, each of capacity Q_e. Among them, V batteries are located on the vehicles and $K - V$ are located at the depot. In each period h, p_h is the energy produced, and it is available at the beginning of the period. If a battery is recharged during a period, it is available only at the end of the period. Moreover, if a vehicle visits the depot to swap its battery, the discharged battery can be recharged only from the following period on. Anyway, vehicles are allowed to visit the depot and wait for the end of the period. The recharging rate is denoted by λ, and corresponds to the maximum quantity of energy that a battery can get in a period. The average state of charge of batteries at the beginning of the time horizon must not be smaller than the average state of charge at the beginning. We make this hypothesis because the idea is to define a sustainable/domestic production and consumption of the energy that vehicles use. In fact, the production unit can be imagined as a solar power plant.

Every transportation request specifies an origin $s_i \in \mathcal{P}$, a destination $t_i \in \mathcal{D}$, a time window $[r_i, d_i]$ and a demand q_i, where $q_{n+i} = -q_i$. Each node in $\mathcal{P} \cup \mathcal{D}$ must be visited exactly once, while the depot may be visited multiple times. The time window to visit the depot is set to $[0, L]$, where L is length of the planning horizon

($L = H\tau$). Moreover, we assume that the number of stops that a vehicle can make to swap its battery is unlimited. Vehicles are allowed to wait at any node in the graph.

At the beginning of each period h, the energy p_h must be split among the batteries located at the depot and the general network. If the energy is assigned to the general network there is a profit. If in a period h the energy produced p_h is not sufficient to recharge the batteries, additional energy can be bought from the general network. Obviously, the energy cannot be bought and sold to the general network in the same period h. The quantity of energy that can be bought from the general network is unlimited. The energy cost is different for each period h: α_h denotes the unitary selling cost and β_h denotes the unitary purchasing cost ($\beta_h > \alpha_h$ for each h). The objective is to satisfy all transportation requests at maximal revenue. The revenue is defined as the difference between the quantity of energy sold to and the quantity of energy bought from the general network. The side-effect is that the energy used by the vehicles must be minimized.

We now introduce the notation used in the problem formulation. We call trip a triple $t = (r, h_1, h_2)$ where r is a path, h_1 is the starting period and h_2 is the ending period of the path. All trips are made up by paths where the depot is the starting and ending node. In what follows, we make the simplifying hypothesis that trips can only start (end) at the beginning (at the end) of a period. We denote the set of trips with $\mathcal{T}$. Let ϵ_{it} denote a parameter that is equal to 1 if request i belongs to trip t and δ_{th} is equal to 1 if trip t is active during period h. c_{th} denote the cost of trip t in period h while γ_k denote the state of charge of battery k at the beginning of the time horizon. For each period h, the continuous variables s_h represent the quantity of energy sold, b_h the quantity of energy bought and l_{kh} the state of charge of battery k at the end of period h. Let a_{kh} denote the quantity of energy given to battery k in period h. u_{kh} is a binary decision variable equal to 1 if battery k is at the depot *for the entire period* h and x_{kt} is a binary decision variable equal to 1 if battery k is used in trip t. Note that if $\sum_{k \in \mathcal{K}} x_{kt} = 0$ then trip t is not selected. The notation is summarized in tables 6.1 and 6.2.

Name	Description
$\mathcal{G} = (\mathcal{N}, \mathcal{A})$	graph
$\mathcal{N}$	set of nodes
$\mathcal{A}$	set of arcs
$\mathcal{P}$	set of pickup nodes
$\mathcal{D}$	set of delivery nodes
0	depot node
$2n+1$	copy of depot node
$\mathcal{R}$	set of requests
s_i	starting station of request i
t_i	ending station of request i
q_i	demand of request i
$[r_i, d_i]$	time window of request i
$\mathcal{H}$	set of periods
τ	period duration
T	planning horizon duration
e_{ij}	cost of arc (i, j)
τ_{ij}	travel time of arc (i, j)
$\mathcal{V}$	set of available vehicles
Q	vehicle capacity
$\mathcal{K}$	set of available batteries
Q_e	battery capacity
p_h	energy produced in period h
λ	battery recharging rate
γ_k	state of charge of battery k at the beginning
α_h	unitary energy selling cost
β_h	unitary energy buying cost
$\mathcal{T}$	set of trips
ϵ_{it}	1 if request i belongs to trip t
δ_{th}	1 if trip t is active during period h
c_{th}	cost of trip t in period h

Table 6.1: Summary of problem parameters

Name	Description
s_h	energy sold in period h
b_h	energy bought in period h
l_{kh}	state of charge of battery k
a_{kh}	energy given to battery k in period h
u_{kh}	1 if battery k is at the depot in period h
x_{kt}	1 if battery k is used in trip t

Table 6.2: Summary of decision variables

The formulation is as follows:

$$\max \sum_{h \in \mathcal{H}} \alpha_h s_h - \sum_{h \in \mathcal{H}} \beta_h b_h \tag{6.1}$$

s.t.:

$$\sum_{k \in \mathcal{K}} a_{kh} + s_h = p_h + b_h \qquad h \in \mathcal{H} \tag{6.2}$$

$$a_{kh} \leq \lambda u_{kh} \qquad k \in \mathcal{K}, h \in \mathcal{H} \tag{6.3}$$

$$l_{k0} = \gamma_k \qquad k \in \mathcal{K} \tag{6.4}$$

$$\sum_{k \in \mathcal{K}} l_{kH} \geq \sum_{k \in \mathcal{K}} l_{k0} \tag{6.5}$$

$$l_{kh} = l_{kh-1} + a_{kh} - \sum_{t \in \mathcal{T}} c_{th} \delta_{th} x_{kt} \qquad k \in \mathcal{K}, h \in \mathcal{H} \tag{6.6}$$

$$l_{kh} \leq Q_e \qquad k \in \mathcal{K}, h \in \mathcal{H} \tag{6.7}$$

$$\sum_{k \in \mathcal{K}} (1 - u_{kh}) \leq V \qquad h \in \mathcal{H} \tag{6.8}$$

$$u_{kh} + \sum_{t \in \mathcal{T}} \delta_{th} x_{kt} \leq 1 \qquad k \in \mathcal{K}, h \in \mathcal{H} \tag{6.9}$$

$$\sum_{t \in \mathcal{T}} \epsilon_{it} \sum_{k \in \mathcal{K}} x_{kt} \geq 1 \qquad i \in \mathcal{R} \tag{6.10}$$

$$b_h, s_h \geq 0 \qquad h \in \mathcal{H} \tag{6.11}$$

$$a_{kh} \geq 0 \qquad k \in \mathcal{K}, h \in \mathcal{H} \tag{6.12}$$

$$l_{kh} \geq 0 \qquad k \in \mathcal{K}, h \in \mathcal{H} \cup \{0\} \tag{6.13}$$

$$u_{kh} \in \{0, 1\} \qquad k \in \mathcal{K}, h \in \mathcal{H} \tag{6.14}$$

$$x_{kt} \in \{0, 1\} \qquad k \in \mathcal{K}, t \in \mathcal{T} \tag{6.15}$$

Constraints (6.2) ensure that in each period, the energy produced is completely split among the batteries and the general network. Constraints (6.3) impose a limit on the amount of energy given to a battery in a period. Constraints (6.4) and (6.5) ensure that the average state of charge at the end of the time horizon is not smaller than the average state of charge at the beginning. Constraints (6.6) model the change in battery charge levels over time. Constraints (6.7) specify the upper bound for the battery charge levels. Constraints (6.8) set an upper bound on the number of active vehicles. Constraints (6.9) ensure that batteries are either at the depot or used by active vehicles. Constraints (6.10) ensure that each request belongs to at least one trip. Constraints (6.11)-(6.15) define the decision variables. Note that time and capacity constraints are taken into account when the trips are generated.

6.4 Solution method

Our problem can be seen as an integrated problem where both routing and energy management decisions have to be made. As formulation (1)-(15) suffers from the fact that the number of feasible trips is exponential, a heuristic algorithm is proposed which is based on a heuristic generation of a subset of feasible trips. Specifically, the solution method is divided into three phases:

1. generation of a subset $\mathcal{T}$ of feasible trips

2. solution of formulation (6.1)-(6.15) based on $\mathcal{T}$

3. repair procedure.

As for phase 1, a randomized construction heuristic is used. Let h_1 and h_2 denote the starting and ending period of a trip t, respectively. For every possible pair (h_1, h_2) a pool of promising trips is generated using a Greedy Randomized Sequential Cheapest Insertion. Algorithms 8 and 9 detail the heuristic insertion procedure.

Algorithm 8 Greedy Randomized Sequential Cheapest Insertion

Input: set of requests $\mathcal{R}$
1: $\mathcal{T} \leftarrow \emptyset$
2: **for all** start periods h_1 in $\mathcal{H}$ **do**
3: **for all** end periods h_2 in $\mathcal{H}$ **do**
4: **while** number of iterations not reached **do**
5: BuildTrips($\mathcal{T}$, h_1, h_2)
6: **end while**
7: **end for**
8: **end for**
Output: $\mathcal{T}$

Algorithm 9 BuildTrips

Input: $\mathcal{T}$, h_1, h_2
1: **while** all feasible requests have been considered **do**
2: choose at random a feasible request and create a new trip
3: **repeat**
4: calculate the insertion cost of all feasible requests not inserted yet
5: choose at random among the 5 best request
6: insert the request in the current trip with probability $\min\{1, \frac{\text{last detour}}{\text{potential detour}}\}$
7: **until** an insertion in the current trip is rejected
8: insert current trip in set $\mathcal{T}$ only if not already present
9: **end while**
Output: $\mathcal{T}$

Algorithm BuildTrips takes as input the start and end periods (h_1 and h_2). Then, only requests that can be inserted in a trip are selected (those that have release dates smaller than h_2 and due dates greater than h_1). All these requests are taken into account in the while loop (line 1 of the algorithm). A request is chosen at random and a new trip is created (line 2). In each iteration of lines 3-7, the cheapest insertion cost of all requests not yet in a trip is calculated and the best request is chosen

from the top 5. The insertion of this request into the current trip is made with a probability based on the insertion cost (called detour). In line 6 of Algorithm 9, last detour denotes the insertion cost of the request inserted in the last iteration, and potential detour denotes the insertion cost of the request chosen in line 5. Then, a random number is drawn from a uniform distribution, and the request is inserted in the current trip only if this number is smaller than the ratio between last detour and potential detour. In line 8, as no more requests can be added in the trip, the trip is added to set $\mathcal{T}$ only if not already present. Two trips $t = (r, h_1, h_2)$ and $t' = (r', h'_1, h'_2)$ are equivalent if and only if $r = r'$, $h_1 = h'_1$ and $h_2 = h'_2$.

Once $\mathcal{T}$ is built formulation (6.1)-(6.15) is solved over set $\mathcal{T}$ (phase 2). Due to constraints (6.10) multiple trips that contain the same request may be selected and therefore a pair of nodes may be visited more than once. In such a case, solutions are repaired as follows (phase 3). If $\mathcal{T}_i$ is the subset of selected trips that all contain request i, i is kept in the trip that corresponds to the minimum insertion cost. Algorithm 10 details the repair procedure.

Algorithm 10 Repair Procedure

Input: $\mathcal{T}$

1: **for all** requests i visited more than once **do**
2: $\quad\mathcal{T}_i \leftarrow$ subset of selected trips that contain i
3: $\quad$**for all** trips t in $\mathcal{T}_i$ **do**
4: $\quad\quad$remove i from t
5: $\quad\quad s(t) = c(t) - c'(t)$
6: $\quad$**end for**
7: $\quad$add again i in the trip t such that $s(t)$ is minimum
8: **end for**

Output: $\mathcal{T}$

In line 5 $c(t)$ and $c'(t)$ denote, respectively, the cost of trip t with and without request i.

6.5 Computational tests

The formulation was tested on benchmark instances introduced in H. Li and Lim, 2003 for the Pickup and Delivery Problem with Time Windows (PDPTW). The instances were adapted by generating data on energy production and consumption, as well as on selling and buying price for energy. Data on the amount of energy produced in each hour of the day were generated consistently with the average irradiance of a typical July day in Naples, Italy. Figure 6.3 shows the average solar irradiation (in $\frac{W}{m^2}$) for

each hour of the day for the month of July for a fixed surface having a 35° slope and 0° azimuth. The data are long-term averages, calculated from hourly global and diffuse irradiance values over the period 2005-2016. The graph was obtained through the use of PVGIS software using data from the PVGIS-SARAH database (PVGIS, 2022).

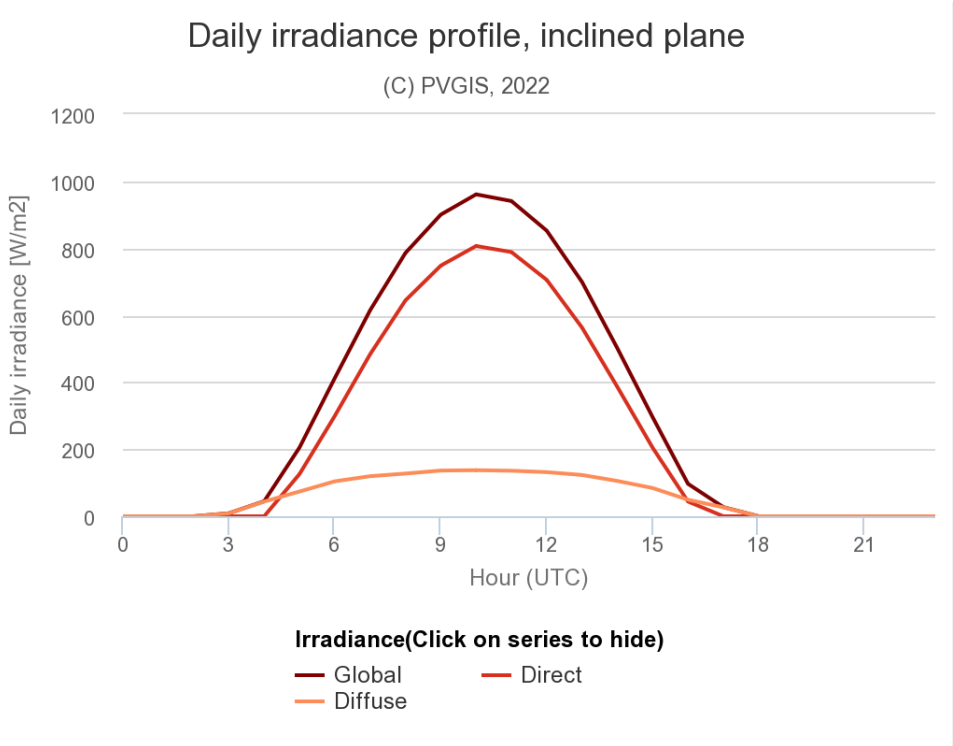


Figure 6.3: Typical irradiance profile of a fixed surface on an average day of July in Naples, Italy

A set of preliminary tests was made with the aim of understanding what are the parameters that mostly affect the difficulty of the instances. Four classes of instances of different size were chosen from the Li & Lim's benchmark instances. An instance with narrow time windows (the instances whose names end in 1) and an instance with large time windows (the instances whose names end in 4) were selected for each class. The number of vehicles and batteries available was set by slightly increasing the number of vehicles from the best known solution to the PDPTW instance. The capacity of the vehicles is set in the instance. The capacity of batteries was set, for each instance, by taking the base value of the cost of the longest trip from the depot to serve a single request, and increasing it by about 1.5.

The value of λ, on the other hand, was set at 118 for almost all instances. Its value can be interpreted as the amount of charge (in minutes) that a battery can

receive in one hour, and it is realistic to think that in one hour a battery can receive a charge equivalent to about two hours of range. This value was kept fixed because in real-world contexts it depends on the characteristics of the charging station. For instances with more than 400 nodes, much larger Q_e and λ values were tested to try to eliminate difficulties related to battery charging.

For the resolution of formulation (6.1)-(6.15), CPLEX 12.9.0 was used. A time limit of 15 minutes was set. Tests were made on a workstation equipped with an Intel Core i7-10610U processor and 16GB of RAM.

Table 6.3 reports the results of the preliminary tests. Column "# nodes" report the number of nodes of the instance. Columns "# trips" and "time(s)" report the number of trips generated by algorithm 8 and the time in second, respectively. Column "V BKS" reports the number of vehicles of the best known solution for the instance of PDPTW. Columns "V av." and "K av." report the number of available vehicles and batteries, respectively, and columns "V us." and "K us." report the number of vehicles and batteries used, respectively. Column "Q" reports the capacity of the vehicles (identical for all instances). Columns "Q_e" and "λ" report the values of the energy-related parameters, *i.e.* the battery capacity and the energy recharging rate. Finally, column "feas" shows whether CPLEX was able to find a feasible solution within the time limit.

instance	# nodes	# trips	time(s)	V BKS	V av.	K av.	V us.	K us.	Q	Q_e	λ	feas
lc101	106	64927	12.25	10	12	12	-	-	200	180	118	no
lc101	106	52131	5.6	10	12	12	12	12	200	180	118	yes
lc104	106	34440	1.1	9	12	12	-	-	200	180	118	no
lc104	106	25798	0.76	9	12	12	11	11	200	180	118	yes
lc1_2_1	212	19382	0.68	20	30	30	-	-	200	425	118	no
lc1_2_1	212	12819	0.34	20	30	30	30	30	200	425	118	yes
lc1_2_4	210	8888	0.32	17	20	25	-	-	200	300	118	no
lc1_2_4	210	8895	0.32	17	25	25	25	25	200	300	118	yes
lc1_2_4	210	8865	0.33	17	20	20	-	-	200	300	118	no
lc1_2_4	210	8860	0.3	17	25	30	-	-	200	300	118	no
lc1_2_4	210	19040	0.94	17	25	25	25	25	200	300	118	yes
lc1_2_4	210	23674	1.24	17	25	25	-	-	200	300	118	no
lr1_4_1	416	6440	0.31	40	60	60	-	-	200	700	118	no
lr1_4_1	416	6520	0.3	40	60	60	-	-	200	1000	1000	no
lr1_4_4	420	9967	3.78	15	25	25	-	-	200	1000	1000	no
lr1_4_4	420	10011	3.87	15	35	35	-	-	200	1000	1000	no
lr1_6_1	634	9531	0.69	59	70	70	-	-	200	1000	1000	no
lr1_6_4	626	15858	6.63	28	40	40	-	-	200	500	500	no

Table 6.3: Preliminary results for Li & Lim instances

The first set of tests conducted were performed to figure out how many trips it is possible to use in solving formulation (6.1)-(6.15). For instances lc101, lc104, lc_1_2_1 and lc1_2_4, the approximate number of trips was found from which formulation (6.1)-(6.15) fails to find a feasible solution in 15 minutes. For instances with more than 400 nodes, CPLEX was unable to find an feasible solution in the time limit, even using a relatively small number of trips generated by Algorithm 8 (the number of trips obtained with 1 iteration in line 4).

Another parameter affecting the difficulty of the instances is the difference between the number of available vehicles and the number of available batteries. It is possible to observe this in the first and fourth rows of instance lc1_2_4. For the instances that cplex was able to solve, all available vehicles and batteries were used, except for instance lc104.

6.6 Conclusions

In this chapter, a new Pickup and Delivery problem with BEVs was introduced. The energy needed to recharge the batteries is produced on-site by a solar-powered Battery Swapping Station (BSS), and the energy management aspect is integrated into the problem. An MILP for the problem and a 3-step resolution scheme that is based on generating a subset of trips were proposed.

Preliminary computational tests were performed on Li & Lim instances for the PDPTW to investigate the impact of parameters in the difficulty of solving the instances. The instances were completed by generating production and energy cost data.

Preliminary computational tests were aimed at studying the parameters of the instances. Analysis of the results of these tests showed that the number of trips used and the size of the instance have an impact on the computation time of a MILP-eligible solution. Other parameters that have an impact on model resolution times are the relative values of the number of available vehicles and batteries: in fact, having more batteries than vehicles makes some instances more difficult to solve.

The perspectives of this work are manifold. The first goal is to complete the study of instance parameters and confirm the results obtained through preliminary tests. In particular, it is interesting to understand what is the impact of the number of batteries on the solutions, as this kind of information can be useful in the infrastructure sizing process.

The second objective is to analyze the solutions of the model as the energy produced and its cost change over the various periods.

Regarding solution methods, another goal is to propose a more efficient solving method. A first step could be to improve the trip generation algorithm. Then the resolution of formulation (6.1)-(6.15) could be improved by applying a Bender's decomposition scheme.

Conclusions and perspectives

In Chapter 2 a new class of Pickup and Delivery problems where the stations are located on circles (rings) was introduced. These problems arise in the field of public transportation systems where autonomous (*i.e.* driverless) vehicles travel on circular networks. These problems are defined on cycle graphs where the nodes represent the stations and the arcs represent the links between pairs of consecutive stations. There is a set of transportation requests, where each request asks for the transportation of a certain quantity from a pickup station to a delivery station. A fleet of capacitated vehicles is available at a particular station (the depot). All problems in this class differ in fleet composition, the presence of time windows associated with the requests and the objective function to optimize. A classification scheme is proposed. Then, the complexity of the variants in which the vehicles are allowed to move in a single direction on the ring and the objective is the minimization of the maximum number of tours is investigated. A polynomial-time algorithm is proposed for some variants and the remaining variants are proven NP-hard. For the NP-hard variants, ILP formulations are proposed and the efficiency of these formulations is evaluated through extensive computational experiments. Experiments on a large number of instances show the efficiency of our formulations. All instances, with up to 160 requests, could be solved in a few minutes. Comparisons with a simple and practically relevant greedy algorithm also confirmed the intrinsic difficulty of the problems/instances and the usefulness of applying exact solution schemes.

Chapter 3 is devoted to the proof of NP-hardness of a problem introduced in Chapter 2. The proof is moved to separate chapter given its length.

In Chapter 4 a special class of PDP-R is studied. The focus is on problems in which vehicles can only travel in a single direction on the ring and the objective is to minimize the sum of the request completion times. All these variants are proven NP-hard. An ILP formulation is proposed and computational tests are executed to evaluate its performance. Results show that the formulation is extremely effective, being able to solve to optimality almost all instances, even the largest ones with 160 requests, in a short computing time. On the most difficult variants of the problem, the formulation is always capable of providing a feasible solution when it exists and largely outperforms a greedy algorithm mimicking common practice. Several techniques to

tackle the hardest instances were explored, with no success at that point.

In Chapter 5 a different class of Pickup and Delivery Problems on rings is studied. In this class of problems, vehicles have the ability to move on the ring following both directions of rotation. The simplest variant in this class of problems, is known in the literature as the Stacker Crane Problem (SCP). Two algorithms for the SCP have already been proposed in the literature, but neither of them is easy to understand and implement. A new simple algorithm for the SCP is proposed. The other problems in this class can be seen as variants of the SCP. These variants are left for future works.

Chapter 6 introduces a new Pickup and Delivery problem in which a set of capacitated Battery Electric Vehicles (BEVs) carry out pickup and delivery operations with time windows constraints. The energy needed to recharge the batteries is produced on-site by a solar-powered Battery Swapping Station (BSS) that is also the depot of the vehicles. Additional batteries are available at the depot, where vehicles can go and swap their batteries. Pickup and delivery operations must be planned over a time horizon divided into periods. In each period it must be decided how much energy to give to the batteries that are at the production unit. Also, if the energy produced is in excess of that required by the batteries, this excess can be sold to the general network at a price. If the energy required by the batteries is greater than the energy produced, an unlimited amount of energy can be bought from the general network. The objective of the problem is to plan vehicle routes to meet all pickup and delivery demands while maximizing the net profit that is made from the energy sold and bought over the time horizon. An MILP formulation of the problem and a matheuristic approach are proposed. The matheuristic approach consists of three steps: in the first one a subset of feasible trips is generated by using a Randomized Construction Heuristic, in the second step the formulation is solved over this set of trips, and in the third one a repair procedure is performed on the obtained solution, in order to avoid more than one trip visiting the same node. Preliminary computational tests are conducted on Li & Lim instances for the PDPTW to investigate the impact of parameters in the difficulty of solving the instances. The instances are completed by generating production and energy cost data. The analysis of the results shows that the number of trips used and the size of the instance have an impact on the computation time of a MILP-eligible solution. Other parameters that have an impact on model resolution times are the relative values of the number of available vehicles and batteries: in fact, having more batteries than vehicles makes some instances more difficult to solve.

This thesis opens the way for multiple research possibilities. The perspectives to which the first part of this thesis opens (i.e., Chapters 2-5) are concerned with exploring further Pickup and Delivery problems on rings. An interesting direction would be to consider different network topologies such as lines or other geometric shapes that can be encountered in practice. Also, autonomous vehicles are bound to use electric engines. A future step of this research could be to investigate the

issues implied by the limited autonomy of electric vehicles (range anxiety, recharging policies...). The classification scheme proposed in Chapter 2 could be enriched to account for new variants where the problem of energy management is considered. Regarding the problems presented in Chapter 4, one perspective is to improve the solving method described in order to solve the most difficult instances. Chapter 5 opens the way for multiple research possibilities. A first perspective is to refine the complexity analysis of the algorithm presented in section 5.3. Another perspective is to complete the complexity analysis of the variants in this class for which the complexity is unknown.

A new Pickup and Delivery problem with electric vehicles is introduced in Chapter 6. The perspectives of this work are manifold. The first goal is to complete the study of instance parameters and confirm the results obtained through preliminary tests. In particular, it would be interesting to understand what is the impact of the number of batteries on the solutions, as this kind of information can be useful in the infrastructure sizing process. The second objective could be to analyze the solutions of the model as the energy produced and its cost change over the various periods. Regarding solution methods, another goal could be to propose a more efficient solving method. A first step could be to improve the trip generation algorithm. Then the resolution of formulation (6.1)-(6.15) could be improved by applying a Bender's decomposition scheme. The proposed scheme also naturally opens the way to an exact method based on column generation, where new trips would be added dynamically to the model.

Appendix A

Detailed computational results of Chapter 2

In section 2.6 we reported the results of the resolution of formulation (2.10)-(2.14). Tables 2.1-2.6 were calculated by taking the average over the 5 instances with the same value of n, V and Q. In this appendix we report the detailed results of those experiments. In all tables, column opt reports if the calculated solution value is optimal for the instance. All other column headings have the same meaning of tables 2.1-2.6.

n	V	Q	instance	CPU(s)	CLT	opt	gapGr(%)
20	1	1	1	0.1	13	yes	7.7
20	1	1	2	0.1	15	yes	6.7
20	1	1	3	0.1	12	yes	8.3
20	1	1	4	0.1	13	yes	0.0
20	1	1	5	0.0	12	yes	0.0
20	1	2	1	0.0	7	yes	14.3
20	1	2	2	0.0	8	yes	12.5
20	1	2	3	0.0	6	yes	0.0
20	1	2	4	0.1	7	yes	0.0
20	1	2	5	0.0	6	yes	0.0
20	1	4	1	0.0	4	yes	25.0
20	1	4	2	0.0	5	yes	0.0
20	1	4	3	0.0	3	yes	33.3
20	1	4	4	0.0	4	yes	0.0
20	1	4	5	0.0	3	yes	0.0
20	1	8	1	0.0	3	yes	0.0
20	1	8	2	0.0	3	yes	0.0
20	1	8	3	0.0	2	yes	50.0
20	1	8	4	0.0	3	yes	0.0
20	1	8	5	0.0	2	yes	50.0
20	2	1	1	0.1	7	yes	14.3
20	2	1	2	0.1	8	yes	12.5
20	2	1	3	0.1	6	yes	16.7
20	2	1	4	0.1	7	yes	0.0
20	2	1	5	0.0	6	yes	0.0
20	2	2	1	0.0	4	yes	25.0
20	2	2	2	0.0	5	yes	0.0
20	2	2	3	0.0	3	yes	33.3
20	2	2	4	0.0	4	yes	0.0
20	2	2	5	0.0	3	yes	0.0
20	2	4	1	0.0	3	yes	0.0
20	2	4	2	0.0	3	yes	0.0
20	2	4	3	0.0	2	yes	50.0
20	2	4	4	0.0	3	yes	0.0
20	2	4	5	0.0	2	yes	50.0
20	2	8	1	0.0	2	yes	0.0
20	2	8	2	0.0	2	yes	0.0
20	2	8	3	0.0	2	yes	0.0
20	2	8	4	0.0	2	yes	0.0
20	2	8	5	0.0	2	yes	0.0
20	3	1	1	0.1	5	yes	20.0
20	3	1	2	0.1	6	yes	0.0
20	3	1	3	0.1	4	yes	25.0
20	3	1	4	0.1	5	yes	0.0
20	3	1	5	0.0	4	yes	0.0
20	3	2	1	0.0	3	yes	33.3
20	3	2	2	0.1	4	yes	0.0
20	3	2	3	0.0	3	yes	0.0
20	3	2	4	0.0	3	yes	0.0
20	3	2	5	0.0	2	yes	0.0
20	3	4	1	0.0	2	yes	50.0
20	3	4	2	0.0	3	yes	0.0
20	3	4	3	0.0	2	yes	0.0
20	3	4	4	0.0	2	yes	0.0
20	3	4	5	0.0	2	yes	0.0
20	3	8	1	0.0	2	yes	0.0
20	3	8	2	0.0	2	yes	0.0
20	3	8	3	0.0	2	yes	0.0
20	3	8	4	0.0	2	yes	0.0
20	3	8	5	0.0	2	yes	0.0

n	V	Q	instance	CPU(s)	CLT	opt	gapGr(%)
40	1	1	1	0.6	21	yes	0.0
40	1	1	2	0.3	25	yes	0.0
40	1	1	3	0.2	29	yes	0.0
40	1	1	4	0.2	27	yes	3.7
40	1	1	5	0.2	25	yes	0.0
40	1	2	1	0.3	11	yes	9.1
40	1	2	2	0.2	13	yes	0.0
40	1	2	3	0.1	15	yes	6.7
40	1	2	4	0.1	14	yes	7.1
40	1	2	5	0.1	13	yes	7.7
40	1	4	1	0.1	6	yes	0.0
40	1	4	2	0.1	7	yes	0.0
40	1	4	3	0.1	8	yes	0.0
40	1	4	4	0.1	7	yes	14.3
40	1	4	5	0.1	7	yes	0.0
40	1	8	1	0.1	4	yes	0.0
40	1	8	2	0.0	4	yes	0.0
40	1	8	3	0.1	4	yes	25.0
40	1	8	4	0.1	4	yes	0.0
40	1	8	5	0.1	4	yes	25.0
40	2	1	1	0.6	11	yes	9.1
40	2	1	2	0.5	13	yes	0.0
40	2	1	3	0.3	15	yes	6.7
40	2	1	4	0.2	14	yes	0.0
40	2	1	5	0.3	13	yes	7.7
40	2	2	1	0.2	6	yes	16.7
40	2	2	2	0.2	7	yes	0.0
40	2	2	3	0.2	8	yes	0.0
40	2	2	4	0.1	7	yes	14.3
40	2	2	5	0.2	7	yes	14.3
40	2	4	1	0.1	4	yes	0.0
40	2	4	2	0.1	4	yes	0.0
40	2	4	3	0.1	4	yes	25.0
40	2	4	4	0.1	4	yes	0.0
40	2	4	5	0.1	4	yes	0.0
40	2	8	1	0.1	3	yes	0.0
40	2	8	2	0.1	3	yes	0.0
40	2	8	3	0.1	3	yes	0.0
40	2	8	4	0.1	2	yes	50.0
40	2	8	5	0.1	2	yes	50.0
40	3	1	1	0.6	8	yes	0.0
40	3	1	2	0.4	9	yes	0.0
40	3	1	3	0.4	10	yes	10.0
40	3	1	4	0.3	9	yes	11.1
40	3	1	5	0.4	9	yes	0.0
40	3	2	1	0.2	5	yes	0.0
40	3	2	2	0.2	5	yes	0.0
40	3	2	3	0.2	5	yes	20.0
40	3	2	4	0.2	5	yes	20.0
40	3	2	5	0.2	5	yes	0.0
40	3	4	1	0.1	3	yes	0.0
40	3	4	2	0.1	3	yes	0.0
40	3	4	3	0.1	3	yes	33.3
40	3	4	4	0.1	3	yes	33.3
40	3	4	5	0.1	3	yes	0.0
40	3	8	1	0.1	2	yes	0.0
40	3	8	2	0.1	2	yes	0.0
40	3	8	3	0.1	2	yes	50.0
40	3	8	4	0.1	2	yes	0.0
40	3	8	5	0.0	2	yes	0.0

Table A.1: Problem $V, Q|sd, u|CLT$ (part 1)

n	V	Q	instance	CPU(s)	CLT	opt	gapGr(%)
80	1	1	1	2.5	45	yes	2.2
80	1	1	2	3.3	44	yes	0.0
80	1	1	3	3.2	48	yes	2.1
80	1	1	4	2.3	49	yes	2.0
80	1	1	5	1.0	46	yes	0.0
80	1	2	1	1.7	23	yes	0.0
80	1	2	2	1.6	22	yes	0.0
80	1	2	3	0.9	24	yes	4.2
80	1	2	4	1.1	25	yes	4.0
80	1	2	5	0.6	23	yes	0.0
80	1	4	1	0.7	12	yes	0.0
80	1	4	2	0.9	11	yes	0.0
80	1	4	3	0.4	12	yes	8.3
80	1	4	4	0.4	13	yes	0.0
80	1	4	5	0.4	12	yes	0.0
80	1	8	1	0.2	6	yes	16.7
80	1	8	2	0.3	6	yes	16.7
80	1	8	3	0.2	7	yes	0.0
80	1	8	4	0.3	7	yes	14.3
80	1	8	5	0.2	6	yes	16.7
80	2	1	1	2.9	23	yes	0.0
80	2	1	2	2.8	22	yes	0.0
80	2	1	3	2.6	24	yes	4.2
80	2	1	4	2.4	25	yes	4.0
80	2	1	5	1.7	23	yes	0.0
80	2	2	1	1.5	12	yes	0.0
80	2	2	2	1.7	11	yes	9.1
80	2	2	3	1.4	12	yes	8.3
80	2	2	4	1.0	13	yes	7.7
80	2	2	5	1.0	12	yes	0.0
80	2	4	1	0.6	6	yes	0.0
80	2	4	2	0.5	6	yes	0.0
80	2	4	3	0.6	7	yes	0.0
80	2	4	4	0.7	7	yes	14.3
80	2	4	5	0.4	6	yes	16.7
80	2	8	1	0.4	3	yes	33.3
80	2	8	2	0.4	3	yes	33.3
80	2	8	3	0.2	4	yes	0.0
80	2	8	4	0.5	4	yes	25.0
80	2	8	5	0.3	3	yes	33.3
80	3	1	1	6.9	15	yes	6.7
80	3	1	2	4.7	15	yes	0.0
80	3	1	3	3.1	16	yes	6.3
80	3	1	4	2.6	17	yes	5.9
80	3	1	5	1.8	16	yes	0.0
80	3	2	1	1.8	8	yes	12.5
80	3	2	2	1.9	8	yes	0.0
80	3	2	3	1.5	8	yes	12.5
80	3	2	4	2.3	9	yes	11.1
80	3	2	5	1.2	8	yes	12.5
80	3	4	1	0.8	4	yes	25.0
80	3	4	2	0.7	4	yes	25.0
80	3	4	3	0.9	5	yes	0.0
80	3	4	4	0.8	5	yes	20.0
80	3	4	5	0.7	4	yes	25.0
80	3	8	1	0.3	3	yes	0.0
80	3	8	2	0.3	3	yes	0.0
80	3	8	3	0.3	3	yes	0.0
80	3	8	4	0.3	3	yes	33.3
80	3	8	5	0.3	3	yes	0.0

n	V	Q	instance	CPU(s)	CLT	opt	gapGr(%)
160	1	1	1	19.2	89	yes	0.0
160	1	1	2	14.3	91	yes	0.0
160	1	1	3	22.6	83	yes	0.0
160	1	1	4	14.2	87	yes	1.1
160	1	1	5	17.5	87	yes	0.0
160	1	2	1	10.5	45	yes	0.0
160	1	2	2	5.4	46	yes	0.0
160	1	2	3	10.0	42	yes	0.0
160	1	2	4	7.7	44	yes	2.3
160	1	2	5	7.4	44	yes	2.3
160	1	4	1	9.4	23	yes	0.0
160	1	4	2	4.1	23	yes	0.0
160	1	4	3	4.2	21	yes	4.8
160	1	4	4	5.7	22	yes	4.5
160	1	4	5	3.0	22	yes	4.5
160	1	8	1	3.0	12	yes	0.0
160	1	8	2	2.6	12	yes	0.0
160	1	8	3	1.5	11	yes	9.1
160	1	8	4	1.6	11	yes	9.1
160	1	8	5	1.4	11	yes	9.1
160	2	1	1	39.8	45	yes	0.0
160	2	1	2	23.1	46	yes	0.0
160	2	1	3	31.6	42	yes	0.0
160	2	1	4	41.4	44	yes	2.3
160	2	1	5	18.5	44	yes	0.0
160	2	2	1	31.3	23	yes	0.0
160	2	2	2	18.3	23	yes	0.0
160	2	2	3	17.1	21	yes	4.8
160	2	2	4	16.8	22	yes	4.5
160	2	2	5	16.2	22	yes	0.0
160	2	4	1	8.2	12	yes	0.0
160	2	4	2	8.0	12	yes	0.0
160	2	4	3	5.0	11	yes	9.1
160	2	4	4	5.8	11	yes	9.1
160	2	4	5	6.0	11	yes	9.1
160	2	8	1	4.5	6	yes	16.7
160	2	8	2	3.3	6	yes	16.7
160	2	8	3	2.7	6	yes	16.7
160	2	8	4	3.9	6	yes	16.7
160	2	8	5	2.4	6	yes	16.7
160	3	1	1	33.5	30	yes	0.0
160	3	1	2	48.1	31	yes	0.0
160	3	1	3	45.0	28	yes	3.6
160	3	1	4	44.5	29	yes	3.4
160	3	1	5	28.5	29	yes	3.4
160	3	2	1	19.0	15	yes	6.7
160	3	2	2	14.1	16	yes	0.0
160	3	2	3	24.9	14	yes	7.1
160	3	2	4	17.6	15	yes	6.7
160	3	2	5	18.5	15	yes	6.7
160	3	4	1	15.4	8	yes	12.5
160	3	4	2	9.8	8	yes	0.0
160	3	4	3	8.7	7	yes	14.3
160	3	4	4	9.7	8	yes	12.5
160	3	4	5	5.7	8	yes	12.5
160	3	8	1	4.8	4	yes	25.0
160	3	8	2	4.7	4	yes	25.0
160	3	8	3	4.4	4	yes	25.0
160	3	8	4	4.3	4	yes	25.0
160	3	8	5	3.4	4	yes	25.0

Table A.2: Problem $V, Q|sd, u|CLT$ (part 2)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
20	1	1	1	0.0	yes	yes	18	-	0.0	yes	yes	18	-	0.0	yes	yes	18	5.6
20	1	1	2	-	no	-	-	-	0.0	yes	yes	16	-	0.0	yes	yes	18	5.6
20	1	1	3	0.0	yes	yes	19	-	-	no	-	-	-	0.0	yes	yes	19	5.3
20	1	1	4	-	no	-	-	-	0.0	yes	yes	19	-	0.0	yes	yes	17	11.8
20	1	1	5	0.0	yes	yes	15	-	0.0	yes	yes	15	6.7	0.0	yes	yes	15	6.7
20	1	2	1	0.0	yes	yes	9	-	0.0	yes	yes	10	-	0.0	yes	yes	9	11.1
20	1	2	2	0.0	yes	yes	10	-	0.0	yes	yes	10	-	0.0	yes	yes	10	10.0
20	1	2	3	0.0	yes	yes	9	-	0.0	yes	yes	8	-	0.0	yes	yes	9	0.0
20	1	2	4	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.0	yes	yes	9	11.1
20	1	2	5	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.0	yes	yes	9	11.1
20	1	4	1	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
20	1	4	2	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
20	1	4	3	0.0	yes	yes	5	-	0.0	yes	yes	4	-	0.0	yes	yes	5	0.0
20	1	4	4	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0	0.0	yes	yes	5	20.0
20	1	4	5	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	0.0
20	1	8	1	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	33.3
20	1	8	2	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	33.3
20	1	8	3	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0	0.0	yes	yes	3	33.3
20	1	8	4	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	33.3
20	1	8	5	0.0	yes	yes	3	-	0.0	yes	yes	2	50.0	0.0	yes	yes	3	0.0
20	2	1	1	0.0	yes	yes	8	-	0.0	yes	yes	10	-	0.0	yes	yes	8	12.5
20	2	1	2	0.0	yes	yes	9	-	0.0	yes	yes	10	-	0.0	yes	yes	9	11.1
20	2	1	3	0.0	yes	yes	9	-	0.0	yes	yes	8	-	0.0	yes	yes	9	0.0
20	2	1	4	0.0	yes	yes	9	-	0.0	yes	yes	10	-	0.0	yes	yes	9	11.1
20	2	1	5	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.0	yes	yes	8	12.5
20	2	2	1	0.0	yes	yes	5	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	2	2	2	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
20	2	2	3	0.0	yes	yes	5	20.0	0.0	yes	yes	5	20.0	0.0	yes	yes	5	20.0
20	2	2	4	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0	0.0	yes	yes	5	20.0
20	2	2	5	0.0	yes	yes	4	25.0	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	2	4	1	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0
20	2	4	2	0.0	yes	yes	3	-	0.0	yes	yes	4	-	0.0	yes	yes	3	33.3
20	2	4	3	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	33.3
20	2	4	4	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	33.3
20	2	4	5	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0
20	2	8	1	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0
20	2	8	2	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0
20	2	8	3	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0
20	2	8	4	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0
20	2	8	5	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0
20	3	1	1	0.0	yes	yes	7	-	0.0	yes	yes	6	-	0.0	yes	yes	7	0.0
20	3	1	2	0.0	yes	yes	7	-	0.0	yes	yes	7	0.0	0.0	yes	yes	7	0.0
20	3	1	3	0.0	yes	yes	6	16.7	0.0	yes	yes	7	0.0	0.0	yes	yes	6	16.7
20	3	1	4	0.0	yes	yes	6	-	-	no	-	-	-	0.0	yes	yes	6	16.7
20	3	1	5	0.0	yes	yes	5	20.0	0.0	yes	yes	6	-	0.0	yes	yes	5	20.0
20	3	2	1	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0	0.0	yes	yes	4	0.0
20	3	2	2	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0	0.0	yes	yes	4	25.0
20	3	2	3	0.0	yes	yes	3	-	0.0	yes	yes	4	-	0.0	yes	yes	3	33.3
20	3	2	4	0.0	yes	yes	4	-	0.0	yes	yes	3	-	0.0	yes	yes	4	25.0
20	3	2	5	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	33.3
20	3	4	1	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0
20	3	4	2	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0
20	3	4	3	0.0	yes	yes	2	-	0.0	yes	yes	3	0.0	0.0	yes	yes	2	50.0
20	3	4	4	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0
20	3	4	5	0.0	yes	yes	3	-	0.0	yes	yes	2	-	0.0	yes	yes	3	0.0
20	3	8	1	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	0.0
20	3	8	2	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0
20	3	8	3	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	0.0
20	3	8	4	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0
20	3	8	5	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0

Table A.3: Problem $V, Q|sd, u, r_i, d_i|CLT$ (part 1)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
40	1	1	1	0.0	yes	yes	33	-	0.0	yes	yes	30	3.3	0.0	yes	yes	33	0.0
40	1	1	2	0.0	yes	yes	34	-	0.0	yes	yes	31	-	0.1	yes	yes	33	6.1
40	1	1	3	0.0	yes	yes	34	-	0.0	yes	yes	32	-	0.1	yes	yes	34	5.9
40	1	1	4	0.0	yes	yes	35	-	0.0	yes	yes	31	-	0.1	yes	yes	35	5.7
40	1	1	5	0.0	yes	yes	35	-	0.0	yes	yes	32	0.0	0.1	yes	yes	35	0.0
40	1	2	1	0.0	yes	yes	16	-	0.0	yes	yes	16	-	0.0	yes	yes	16	6.3
40	1	2	2	0.0	yes	yes	17	-	0.0	yes	yes	17	-	0.0	yes	yes	17	5.9
40	1	2	3	0.0	yes	yes	17	-	0.0	yes	yes	18	-	0.0	yes	yes	17	5.9
40	1	2	4	0.0	yes	yes	15	-	0.0	yes	yes	16	-	0.0	yes	yes	15	13.3
40	1	2	5	0.0	yes	yes	17	-	0.0	yes	yes	16	-	0.0	yes	yes	17	5.9
40	1	4	1	0.0	yes	yes	10	-	0.0	yes	yes	9	-	0.0	yes	yes	10	0.0
40	1	4	2	0.0	yes	yes	9	-	0.0	yes	yes	8	-	0.0	yes	yes	9	11.1
40	1	4	3	0.0	yes	yes	9	-	0.0	yes	yes	8	-	0.0	yes	yes	8	12.5
40	1	4	4	0.0	yes	yes	9	-	0.0	yes	yes	9	0.0	0.0	yes	yes	9	0.0
40	1	4	5	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.0	yes	yes	9	0.0
40	1	8	1	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	0.0
40	1	8	2	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
40	1	8	3	0.0	yes	yes	5	-	0.0	yes	yes	4	-	0.0	yes	yes	5	0.0
40	1	8	4	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	0.0
40	1	8	5	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
40	2	1	1	0.0	yes	yes	17	-	0.0	yes	yes	16	-	0.1	yes	yes	17	0.0
40	2	1	2	0.0	yes	yes	18	-	0.0	yes	yes	17	-	0.1	yes	yes	18	0.0
40	2	1	3	0.0	yes	yes	16	-	0.0	yes	yes	17	-	0.2	yes	yes	16	12.5
40	2	1	4	0.0	yes	yes	16	-	0.0	yes	yes	16	-	0.0	yes	yes	16	0.0
40	2	1	5	0.0	yes	yes	16	-	0.0	yes	yes	16	-	0.1	yes	yes	16	6.3
40	2	2	1	0.0	yes	yes	9	-	0.0	yes	yes	9	11.1	0.0	yes	yes	9	0.0
40	2	2	2	0.0	yes	yes	9	-	0.0	yes	yes	9	0.0	0.0	yes	yes	9	0.0
40	2	2	3	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.1	yes	yes	9	11.1
40	2	2	4	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.0	yes	yes	9	0.0
40	2	2	5	0.0	yes	yes	9	0.0	0.0	yes	yes	8	12.5	0.0	yes	yes	9	0.0
40	2	4	1	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	0.0
40	2	4	2	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
40	2	4	3	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
40	2	4	4	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	0.0
40	2	4	5	0.0	yes	yes	4	-	0.0	yes	yes	5	-	0.0	yes	yes	4	25.0
40	2	8	1	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	33.3
40	2	8	2	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	33.3
40	2	8	3	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	33.3
40	2	8	4	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	33.3
40	2	8	5	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	33.3
40	3	1	1	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.1	yes	yes	11	0.0
40	3	1	2	0.0	yes	yes	11	-	0.0	yes	yes	12	-	0.1	yes	yes	11	9.1
40	3	1	3	0.0	yes	yes	11	-	0.0	yes	yes	12	8.3	0.2	yes	yes	11	9.1
40	3	1	4	0.0	yes	yes	10	-	0.0	yes	yes	11	-	0.1	yes	yes	10	10.0
40	3	1	5	0.0	yes	yes	12	-	0.0	yes	yes	12	-	0.1	yes	yes	12	8.3
40	3	2	1	0.0	yes	yes	7	-	0.0	yes	yes	6	-	0.1	yes	yes	7	14.3
40	3	2	2	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.1	yes	yes	6	16.7
40	3	2	3	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.1	yes	yes	6	16.7
40	3	2	4	0.0	yes	yes	6	-	0.0	yes	yes	5	-	0.1	yes	yes	6	33.3
40	3	2	5	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	0.0
40	3	4	1	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
40	3	4	2	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
40	3	4	3	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.1	yes	yes	4	0.0
40	3	4	4	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.1	yes	yes	4	0.0
40	3	4	5	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
40	3	8	1	0.0	yes	yes	3	-	0.1	yes	yes	3	-	0.1	yes	yes	3	0.0
40	3	8	2	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.1	yes	yes	3	0.0
40	3	8	3	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.1	yes	yes	3	0.0
40	3	8	4	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.1	yes	yes	3	0.0
40	3	8	5	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.1	yes	yes	3	0.0

Table A.4: Problem $V, Q|sd, u, r_i, d_i|CLT$ (part 2)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
80	1	1	1	0.1	yes	yes	62	-	0.1	yes	yes	63	-	0.2	yes	yes	62	1.6
80	1	1	2	0.1	yes	yes	61	-	0.1	yes	yes	60	-	0.2	yes	yes	61	1.6
80	1	1	3	0.2	yes	yes	63	3.2	0.2	yes	yes	63	-	0.4	yes	yes	63	1.6
80	1	1	4	0.1	yes	yes	61	-	-	no	-	-	-	0.1	yes	yes	61	0.0
80	1	1	5	0.1	yes	yes	60	-	0.1	yes	yes	62	-	0.1	yes	yes	60	1.7
80	1	2	1	0.0	yes	yes	31	-	0.0	yes	yes	32	-	0.1	yes	yes	31	3.2
80	1	2	2	0.0	yes	yes	31	-	0.0	yes	yes	32	-	0.1	yes	yes	31	3.2
80	1	2	3	0.0	yes	yes	32	-	0.1	yes	yes	33	-	0.1	yes	yes	32	0.0
80	1	2	4	0.0	yes	yes	32	-	0.0	yes	yes	31	-	0.2	yes	yes	32	3.1
80	1	2	5	0.0	yes	yes	31	-	0.0	yes	yes	30	-	0.1	yes	yes	31	3.2
80	1	4	1	0.0	yes	yes	17	-	0.0	yes	yes	15	-	0.1	yes	yes	17	5.9
80	1	4	2	0.0	yes	yes	17	-	0.0	yes	yes	15	-	0.1	yes	yes	17	0.0
80	1	4	3	0.0	yes	yes	17	-	0.0	yes	yes	16	-	0.1	yes	yes	17	0.0
80	1	4	4	0.0	yes	yes	16	-	0.0	yes	yes	16	6.3	0.1	yes	yes	16	12.5
80	1	4	5	0.0	yes	yes	16	6.3	0.0	yes	yes	16	-	0.1	yes	yes	16	6.3
80	1	8	1	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.1	yes	yes	9	11.1
80	1	8	2	0.0	yes	yes	9	-	0.0	yes	yes	9	0.0	0.1	yes	yes	9	11.1
80	1	8	3	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.1	yes	yes	9	0.0
80	1	8	4	0.0	yes	yes	9	-	0.0	yes	yes	8	-	0.1	yes	yes	9	11.1
80	1	8	5	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.0	yes	yes	9	11.1
80	2	1	1	0.1	yes	yes	31	-	0.2	yes	yes	31	-	0.2	yes	yes	31	3.2
80	2	1	2	0.0	yes	yes	31	0.0	0.1	yes	yes	31	-	0.3	yes	yes	31	0.0
80	2	1	3	0.1	yes	yes	32	-	0.2	yes	yes	32	-	0.6	yes	yes	32	3.1
80	2	1	4	0.1	yes	yes	32	-	0.1	yes	yes	31	-	0.3	yes	yes	32	0.0
80	2	1	5	0.1	yes	yes	31	-	0.1	yes	yes	30	-	0.3	yes	yes	31	6.5
80	2	2	1	0.0	yes	yes	16	-	0.0	yes	yes	16	-	0.2	yes	yes	16	0.0
80	2	2	2	0.0	yes	yes	16	-	0.0	yes	yes	17	-	0.1	yes	yes	16	6.3
80	2	2	3	0.0	yes	yes	16	-	0.0	yes	yes	16	-	0.3	yes	yes	16	6.3
80	2	2	4	0.0	yes	yes	16	-	0.0	yes	yes	16	-	0.1	yes	yes	16	6.3
80	2	2	5	0.0	yes	yes	15	-	0.0	yes	yes	16	6.3	0.2	yes	yes	15	6.7
80	2	4	1	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.2	yes	yes	9	0.0
80	2	4	2	0.0	yes	yes	8	12.5	0.0	yes	yes	9	-	0.2	yes	yes	8	12.5
80	2	4	3	0.0	yes	yes	8	-	0.0	yes	yes	9	-	0.2	yes	yes	8	12.5
80	2	4	4	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.2	yes	yes	9	11.1
80	2	4	5	0.0	yes	yes	9	0.0	0.0	yes	yes	8	-	0.1	yes	yes	9	0.0
80	2	8	1	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.2	yes	yes	5	20.0
80	2	8	2	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.2	yes	yes	5	0.0
80	2	8	3	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.2	yes	yes	5	20.0
80	2	8	4	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.1	yes	yes	5	20.0
80	2	8	5	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.1	yes	yes	5	20.0
80	3	1	1	0.1	yes	yes	21	-	0.1	yes	yes	21	-	0.4	yes	yes	21	4.8
80	3	1	2	0.1	yes	yes	22	-	0.2	yes	yes	22	4.5	0.5	yes	yes	22	0.0
80	3	1	3	0.1	yes	yes	22	-	0.2	yes	yes	22	-	0.5	yes	yes	22	0.0
80	3	1	4	0.1	yes	yes	21	-	0.2	yes	yes	21	-	0.5	yes	yes	21	9.5
80	3	1	5	0.0	yes	yes	21	4.8	0.1	yes	yes	21	-	0.3	yes	yes	21	4.8
80	3	2	1	0.0	yes	yes	11	-	0.1	yes	yes	11	-	0.3	yes	yes	11	9.1
80	3	2	2	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.3	yes	yes	11	9.1
80	3	2	3	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.3	yes	yes	11	9.1
80	3	2	4	0.0	yes	yes	11	0.0	0.1	yes	yes	11	-	0.3	yes	yes	11	0.0
80	3	2	5	0.0	yes	yes	11	-	0.0	yes	yes	11	9.1	0.2	yes	yes	11	9.1
80	3	4	1	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.2	yes	yes	6	16.7
80	3	4	2	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.2	yes	yes	6	16.7
80	3	4	3	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.3	yes	yes	6	16.7
80	3	4	4	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.2	yes	yes	6	16.7
80	3	4	5	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.2	yes	yes	6	16.7
80	3	8	1	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.2	yes	yes	4	0.0
80	3	8	2	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.2	yes	yes	4	25.0
80	3	8	3	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.2	yes	yes	4	0.0
80	3	8	4	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.3	yes	yes	4	25.0
80	3	8	5	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.2	yes	yes	4	25.0

Table A.5: Problem $V, Q|sd, u, r_i, d_i|CLT$ (part 3)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
160	1	1	1	0.7	yes	yes	119	-	-	no	-	-	-	0.7	yes	yes	119	0.8
160	1	1	2	0.9	yes	yes	122	-	0.8	yes	yes	123	-	4.8	yes	yes	122	1.6
160	1	1	3	0.9	yes	yes	119	-	1.3	yes	yes	123	-	1.9	yes	yes	119	3.4
160	1	1	4	0.2	yes	yes	122	-	0.8	yes	yes	119	-	1.0	yes	yes	122	1.6
160	1	1	5	0.4	yes	yes	121	-	0.9	yes	yes	122	-	0.9	yes	yes	121	0.0
160	1	2	1	0.1	yes	yes	61	1.6	0.3	yes	yes	62	-	0.5	yes	yes	61	1.6
160	1	2	2	0.1	yes	yes	62	-	0.1	yes	yes	61	-	0.5	yes	yes	62	3.2
160	1	2	3	0.1	yes	yes	61	-	0.1	yes	yes	61	-	0.5	yes	yes	61	0.0
160	1	2	4	0.1	yes	yes	61	-	0.1	yes	yes	61	1.6	0.5	yes	yes	61	0.0
160	1	2	5	0.1	yes	yes	63	-	0.1	yes	yes	61	-	0.4	yes	yes	63	0.0
160	1	4	1	0.0	yes	yes	31	-	0.1	yes	yes	31	-	0.3	yes	yes	31	3.2
160	1	4	2	0.0	yes	yes	30	3.3	0.1	yes	yes	31	-	0.4	yes	yes	30	3.3
160	1	4	3	0.0	yes	yes	31	-	0.1	yes	yes	30	-	0.4	yes	yes	31	0.0
160	1	4	4	0.0	yes	yes	31	-	0.1	yes	yes	32	-	0.3	yes	yes	31	0.0
160	1	4	5	0.0	yes	yes	31	-	0.1	yes	yes	31	-	0.3	yes	yes	31	3.2
160	1	8	1	0.0	yes	yes	16	-	0.0	yes	yes	16	0.0	0.3	yes	yes	16	6.3
160	1	8	2	0.0	yes	yes	16	-	0.0	yes	yes	16	-	0.3	yes	yes	16	6.3
160	1	8	3	0.0	yes	yes	16	-	0.0	yes	yes	16	6.3	0.3	yes	yes	16	6.3
160	1	8	4	0.0	yes	yes	16	-	0.0	yes	yes	16	-	0.3	yes	yes	16	6.3
160	1	8	5	0.0	yes	yes	16	-	0.0	yes	yes	16	-	0.2	yes	yes	16	0.0
160	2	1	1	0.5	yes	yes	63	-	1.1	yes	yes	62	-	2.3	yes	yes	63	1.6
160	2	1	2	0.4	yes	yes	62	-	0.8	yes	yes	60	-	1.7	yes	yes	62	0.0
160	2	1	3	1.2	yes	yes	61	-	0.8	yes	yes	61	-	1.8	yes	yes	61	0.0
160	2	1	4	0.7	yes	yes	64	-	0.7	yes	yes	61	-	3.0	yes	yes	64	1.6
160	2	1	5	0.5	yes	yes	61	-	0.3	yes	yes	61	-	2.1	yes	yes	61	1.6
160	2	2	1	0.1	yes	yes	31	-	0.3	yes	yes	32	-	0.9	yes	yes	31	0.0
160	2	2	2	0.1	yes	yes	31	-	0.2	yes	yes	31	-	0.9	yes	yes	31	3.2
160	2	2	3	0.1	yes	yes	32	-	0.1	yes	yes	31	-	1.0	yes	yes	32	0.0
160	2	2	4	0.1	yes	yes	31	-	0.1	yes	yes	32	0.0	1.0	yes	yes	31	6.5
160	2	2	5	0.1	yes	yes	31	-	0.1	yes	yes	32	-	0.6	yes	yes	31	3.2
160	2	4	1	0.0	yes	yes	16	-	0.1	yes	yes	16	-	0.7	yes	yes	16	0.0
160	2	4	2	0.0	yes	yes	16	6.3	0.1	yes	yes	16	-	0.7	yes	yes	16	6.3
160	2	4	3	0.0	yes	yes	17	-	0.0	yes	yes	16	-	0.8	yes	yes	17	0.0
160	2	4	4	0.0	yes	yes	16	-	0.1	yes	yes	16	-	0.7	yes	yes	16	6.3
160	2	4	5	0.0	yes	yes	16	-	0.1	yes	yes	17	-	0.7	yes	yes	16	6.3
160	2	8	1	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.7	yes	yes	9	11.1
160	2	8	2	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.7	yes	yes	9	11.1
160	2	8	3	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.7	yes	yes	9	0.0
160	2	8	4	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.7	yes	yes	9	0.0
160	2	8	5	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.7	yes	yes	9	0.0
160	3	1	1	0.6	yes	yes	42	-	0.6	yes	yes	42	-	2.1	yes	yes	42	0.0
160	3	1	2	0.6	yes	yes	41	-	0.8	yes	yes	41	-	1.9	yes	yes	41	0.0
160	3	1	3	0.4	yes	yes	42	-	0.3	yes	yes	40	2.5	2.2	yes	yes	42	0.0
160	3	1	4	0.6	yes	yes	41	-	0.8	yes	yes	41	-	1.8	yes	yes	41	2.4
160	3	1	5	0.3	yes	yes	42	-	0.3	yes	yes	40	-	2.3	yes	yes	42	0.0
160	3	2	1	0.2	yes	yes	21	-	0.3	yes	yes	22	-	2.7	yes	yes	21	4.8
160	3	2	2	0.1	yes	yes	21	4.8	0.2	yes	yes	21	-	1.2	yes	yes	21	4.8
160	3	2	3	0.1	yes	yes	20	-	0.1	yes	yes	21	-	0.8	yes	yes	20	5.0
160	3	2	4	0.1	yes	yes	21	-	0.2	yes	yes	21	4.8	1.6	yes	yes	21	4.8
160	3	2	5	0.1	yes	yes	21	-	0.1	yes	yes	21	-	1.2	yes	yes	21	4.8
160	3	4	1	0.0	yes	yes	11	-	0.1	yes	yes	11	-	1.1	yes	yes	11	9.1
160	3	4	2	0.1	yes	yes	11	-	0.1	yes	yes	11	-	1.6	yes	yes	11	9.1
160	3	4	3	0.0	yes	yes	11	-	0.1	yes	yes	11	9.1	1.1	yes	yes	11	0.0
160	3	4	4	0.1	yes	yes	12	-	0.1	yes	yes	11	-	1.1	yes	yes	12	0.0
160	3	4	5	0.1	yes	yes	11	-	0.1	yes	yes	11	-	1.1	yes	yes	11	9.1
160	3	8	1	0.0	yes	yes	6	-	0.0	yes	yes	6	-	1.1	yes	yes	6	16.7
160	3	8	2	0.0	yes	yes	6	-	0.0	yes	yes	6	-	1.1	yes	yes	6	16.7
160	3	8	3	0.0	yes	yes	6	-	0.0	yes	yes	6	-	1.0	yes	yes	6	16.7
160	3	8	4	0.0	yes	yes	6	-	0.0	yes	yes	6	-	1.1	yes	yes	6	16.7
160	3	8	5	0.0	yes	yes	6	-	0.0	yes	yes	6	-	1.0	yes	yes	6	16.7

Table A.6: Problem $V, Q|sd, u, r_i, d_i|CLT$ (part 4)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
20	1	1	1	0.0	yes	yes	22	-	0.0	yes	yes	20	-	0.0	yes	yes	22	4.5
20	1	1	2	0.0	yes	yes	22	-	0.0	yes	yes	21	-	0.0	yes	yes	22	4.5
20	1	1	3	0.0	yes	yes	19	-	0.0	yes	yes	18	-	0.0	yes	yes	19	10.5
20	1	1	4	0.0	yes	yes	20	5.0	0.0	yes	yes	20	-	0.0	yes	yes	20	0.0
20	1	1	5	0.0	yes	yes	20	5.0	0.0	yes	yes	18	5.6	0.0	yes	yes	20	5.0
20	1	2	1	0.0	yes	yes	12	-	0.0	yes	yes	11	-	0.0	yes	yes	11	9.1
20	1	2	2	0.0	yes	yes	12	8.3	0.0	yes	yes	11	-	0.0	yes	yes	12	8.3
20	1	2	3	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.0	yes	yes	10	10.0
20	1	2	4	0.0	yes	yes	11	-	0.0	yes	yes	11	0.0	0.0	yes	yes	11	9.1
20	1	2	5	0.0	yes	yes	10	-	0.0	yes	yes	9	-	0.0	yes	yes	10	10.0
20	1	4	1	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	0.0
20	1	4	2	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7	0.0	yes	yes	6	16.7
20	1	4	3	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
20	1	4	4	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
20	1	4	5	0.0	yes	yes	6	16.7	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
20	1	8	1	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
20	1	8	2	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	1	8	3	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
20	1	8	4	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	3	33.3
20	1	8	5	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
20	2	1	1	0.0	yes	yes	11	-	0.0	yes	yes	10	10.0	0.0	yes	yes	11	0.0
20	2	1	2	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.0	yes	yes	11	0.0
20	2	1	3	0.0	yes	yes	10	-	0.0	yes	yes	11	0.0	0.0	yes	yes	10	10.0
20	2	1	4	0.0	yes	yes	11	-	0.0	yes	yes	11	9.1	0.0	yes	yes	11	0.0
20	2	1	5	0.0	yes	yes	10	10.0	0.0	yes	yes	11	9.1	0.0	yes	yes	10	10.0
20	2	2	1	0.0	yes	yes	6	16.7	0.0	yes	yes	6	16.7	0.0	yes	yes	6	16.7
20	2	2	2	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
20	2	2	3	0.0	yes	yes	6	-	0.0	yes	yes	5	20.0	0.0	yes	yes	6	16.7
20	2	2	4	0.0	yes	yes	6	-	0.0	yes	yes	6	0.0	0.0	yes	yes	6	0.0
20	2	2	5	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	0.0
20	2	4	1	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
20	2	4	2	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	2	4	3	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	2	4	4	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
20	2	4	5	0.0	yes	yes	4	-	0.0	yes	yes	3	-	0.0	yes	yes	4	0.0
20	2	8	1	0.0	yes	yes	3	-	0.0	yes	yes	2	-	0.0	yes	yes	3	0.0
20	2	8	2	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0
20	2	8	3	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0
20	2	8	4	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0
20	2	8	5	0.0	yes	yes	3	-	0.0	yes	yes	2	-	0.0	yes	yes	3	0.0
20	3	1	1	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.0	yes	yes	8	0.0
20	3	1	2	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.0	yes	yes	8	12.5
20	3	1	3	0.0	yes	yes	8	0.0	0.0	yes	yes	8	-	0.0	yes	yes	8	0.0
20	3	1	4	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.0	yes	yes	8	12.5
20	3	1	5	0.0	yes	yes	7	-	0.0	yes	yes	8	0.0	0.0	yes	yes	7	14.3
20	3	2	1	0.0	yes	yes	4	-	0.0	yes	yes	5	0.0	0.0	yes	yes	4	25.0
20	3	2	2	0.0	yes	yes	5	-	0.0	yes	yes	4	-	0.0	yes	yes	5	0.0
20	3	2	3	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	50.0
20	3	2	4	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	0.0
20	3	2	5	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	3	4	1	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0
20	3	4	2	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0
20	3	4	3	0.0	yes	yes	2	-	0.0	yes	yes	3	-	0.0	yes	yes	2	50.0
20	3	4	4	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0
20	3	4	5	0.0	yes	yes	2	-	0.0	yes	yes	3	-	0.0	yes	yes	2	50.0
20	3	8	1	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0
20	3	8	2	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0
20	3	8	3	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0
20	3	8	4	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0
20	3	8	5	0.0	yes	yes	2	-	0.0	yes	yes	2	-	0.0	yes	yes	2	50.0

Table A.7: Problem $V, Q|sd, u, r_i, d_i|CLT$ (part 5)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
40	1	1	1	0.0	yes	yes	39	-	0.0	yes	yes	44	-	0.0	yes	yes	39	2.6
40	1	1	2	0.0	yes	yes	40	-	0.0	yes	yes	40	-	0.0	yes	yes	40	2.5
40	1	1	3	0.0	yes	yes	38	-	0.0	yes	yes	39	-	0.0	yes	yes	38	2.6
40	1	1	4	0.0	yes	yes	38	-	0.0	yes	yes	44	-	0.0	yes	yes	38	0.0
40	1	1	5	0.0	yes	yes	41	-	0.0	yes	yes	40	-	0.0	yes	yes	41	4.9
40	1	2	1	0.0	yes	yes	21	-	0.0	yes	yes	21	-	0.0	yes	yes	21	4.8
40	1	2	2	0.0	yes	yes	21	-	0.0	yes	yes	20	-	0.0	yes	yes	21	4.8
40	1	2	3	0.0	yes	yes	21	-	0.0	yes	yes	21	-	0.0	yes	yes	21	4.8
40	1	2	4	0.0	yes	yes	20	-	0.0	yes	yes	21	4.8	0.0	yes	yes	20	10.0
40	1	2	5	0.0	yes	yes	20	-	0.0	yes	yes	21	-	0.0	yes	yes	20	0.0
40	1	4	1	0.0	yes	yes	11	-	0.0	yes	yes	10	10.0	0.0	yes	yes	11	0.0
40	1	4	2	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.0	yes	yes	11	0.0
40	1	4	3	0.0	yes	yes	11	-	0.0	yes	yes	12	-	0.0	yes	yes	11	9.1
40	1	4	4	0.0	yes	yes	11	-	0.0	yes	yes	10	-	0.0	yes	yes	11	0.0
40	1	4	5	0.0	yes	yes	11	9.1	0.0	yes	yes	11	0.0	0.0	yes	yes	11	9.1
40	1	8	1	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7	0.0	yes	yes	6	16.7
40	1	8	2	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	0.0
40	1	8	3	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
40	1	8	4	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
40	1	8	5	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
40	2	1	1	0.0	yes	yes	19	-	0.0	yes	yes	20	-	0.0	yes	yes	19	5.3
40	2	1	2	0.0	yes	yes	21	-	0.0	yes	yes	21	4.8	0.1	yes	yes	21	4.8
40	2	1	3	0.0	yes	yes	21	4.8	0.0	yes	yes	20	-	0.1	yes	yes	21	4.8
40	2	1	4	0.0	yes	yes	21	-	0.0	yes	yes	20	-	0.0	yes	yes	21	4.8
40	2	1	5	0.0	yes	yes	22	-	0.0	yes	yes	21	-	0.1	yes	yes	22	0.0
40	2	2	1	0.0	yes	yes	10	-	0.0	yes	yes	11	-	0.0	yes	yes	10	10.0
40	2	2	2	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.0	yes	yes	11	0.0
40	2	2	3	0.0	yes	yes	11	-	0.0	yes	yes	11	9.1	0.0	yes	yes	11	0.0
40	2	2	4	0.0	yes	yes	10	-	0.0	yes	yes	10	-	0.0	yes	yes	10	10.0
40	2	2	5	0.0	yes	yes	11	-	0.0	yes	yes	10	10.0	0.0	yes	yes	11	0.0
40	2	4	1	0.0	yes	yes	6	16.7	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
40	2	4	2	0.0	yes	yes	6	-	0.0	yes	yes	6	0.0	0.0	yes	yes	6	16.7
40	2	4	3	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
40	2	4	4	0.0	yes	yes	6	0.0	0.0	yes	yes	6	-	0.0	yes	yes	6	0.0
40	2	4	5	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
40	2	8	1	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
40	2	8	2	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
40	2	8	3	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
40	2	8	4	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
40	2	8	5	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0
40	3	1	1	0.0	yes	yes	15	0.0	0.0	yes	yes	15	-	0.1	yes	yes	15	0.0
40	3	1	2	0.0	yes	yes	15	-	0.0	yes	yes	14	-	0.1	yes	yes	15	0.0
40	3	1	3	0.0	yes	yes	14	-	0.0	yes	yes	14	0.0	0.1	yes	yes	14	0.0
40	3	1	4	0.0	yes	yes	14	-	0.0	yes	yes	15	-	0.1	yes	yes	14	7.1
40	3	1	5	0.0	yes	yes	15	-	0.0	yes	yes	14	-	0.1	yes	yes	15	6.7
40	3	2	1	0.0	yes	yes	8	-	0.0	yes	yes	8	12.5	0.1	yes	yes	8	12.5
40	3	2	2	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.0	yes	yes	8	0.0
40	3	2	3	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.0	yes	yes	8	0.0
40	3	2	4	0.0	yes	yes	8	-	0.0	yes	yes	7	-	0.0	yes	yes	8	0.0
40	3	2	5	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.1	yes	yes	8	0.0
40	3	4	1	0.0	yes	yes	4	-	0.0	yes	yes	5	-	0.1	yes	yes	4	25.0
40	3	4	2	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.1	yes	yes	5	0.0
40	3	4	3	0.0	yes	yes	4	-	0.0	yes	yes	5	0.0	0.0	yes	yes	4	0.0
40	3	4	4	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
40	3	4	5	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.1	yes	yes	5	0.0
40	3	8	1	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.1	yes	yes	3	33.3
40	3	8	2	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.1	yes	yes	3	0.0
40	3	8	3	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.1	yes	yes	3	0.0
40	3	8	4	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.0	yes	yes	3	0.0
40	3	8	5	0.0	yes	yes	3	-	0.0	yes	yes	3	-	0.1	yes	yes	3	0.0

Table A.8: Problem $V, Q|sd, u, r_i, d_i|CLT$ (part 6)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
80	1	1	1	0.1	yes	yes	80	1.3	0.1	yes	yes	82	1.2	0.1	yes	yes	80	1.3
80	1	1	2	0.0	yes	yes	80	-	0.1	yes	yes	80	2.5	0.1	yes	yes	80	0.0
80	1	1	3	0.1	yes	yes	79	-	0.1	yes	yes	78	0.0	0.1	yes	yes	79	0.0
80	1	1	4	0.1	yes	yes	80	-	0.1	yes	yes	82	1.2	0.2	yes	yes	80	2.5
80	1	1	5	0.0	yes	yes	82	0.0	0.1	yes	yes	81	-	0.1	yes	yes	82	0.0
80	1	2	1	0.0	yes	yes	40	-	0.0	yes	yes	41	0.0	0.1	yes	yes	40	0.0
80	1	2	2	0.0	yes	yes	40	-	0.0	yes	yes	41	0.0	0.1	yes	yes	40	2.5
80	1	2	3	0.0	yes	yes	41	0.0	0.0	yes	yes	42	2.4	0.1	yes	yes	41	0.0
80	1	2	4	0.0	yes	yes	41	-	0.0	yes	yes	40	2.5	0.1	yes	yes	41	2.4
80	1	2	5	0.0	yes	yes	40	-	0.0	yes	yes	41	-	0.1	yes	yes	40	2.5
80	1	4	1	0.0	yes	yes	20	-	0.0	yes	yes	21	-	0.1	yes	yes	20	5.0
80	1	4	2	0.0	yes	yes	21	-	0.0	yes	yes	20	-	0.1	yes	yes	21	4.8
80	1	4	3	0.0	yes	yes	20	-	0.0	yes	yes	22	0.0	0.1	yes	yes	20	5.0
80	1	4	4	0.0	yes	yes	20	-	0.0	yes	yes	21	-	0.1	yes	yes	20	5.0
80	1	4	5	0.0	yes	yes	21	-	0.0	yes	yes	20	-	0.1	yes	yes	21	4.8
80	1	8	1	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.1	yes	yes	11	0.0
80	1	8	2	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.1	yes	yes	11	9.1
80	1	8	3	0.0	yes	yes	11	-	0.0	yes	yes	11	9.1	0.1	yes	yes	11	9.1
80	1	8	4	0.0	yes	yes	10	-	0.0	yes	yes	11	-	0.0	yes	yes	10	10.0
80	1	8	5	0.0	yes	yes	10	-	0.0	yes	yes	11	-	0.1	yes	yes	10	10.0
80	2	1	1	0.0	yes	yes	40	-	0.1	yes	yes	41	-	0.2	yes	yes	40	2.5
80	2	1	2	0.1	yes	yes	42	-	0.1	yes	yes	41	-	0.5	yes	yes	42	2.4
80	2	1	3	0.1	yes	yes	41	-	0.1	yes	yes	40	-	0.4	yes	yes	41	0.0
80	2	1	4	0.1	yes	yes	41	-	0.1	yes	yes	41	0.0	0.3	yes	yes	41	2.4
80	2	1	5	0.1	yes	yes	41	4.9	0.1	yes	yes	41	2.4	0.3	yes	yes	41	2.4
80	2	2	1	0.0	yes	yes	21	-	0.0	yes	yes	21	-	0.2	yes	yes	21	4.8
80	2	2	2	0.0	yes	yes	21	-	0.0	yes	yes	21	0.0	0.2	yes	yes	21	0.0
80	2	2	3	0.0	yes	yes	20	-	0.0	yes	yes	21	-	0.2	yes	yes	20	5.0
80	2	2	4	0.0	yes	yes	21	-	0.0	yes	yes	21	-	0.1	yes	yes	21	4.8
80	2	2	5	0.0	yes	yes	22	0.0	0.0	yes	yes	20	5.0	0.2	yes	yes	22	0.0
80	2	4	1	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.1	yes	yes	11	9.1
80	2	4	2	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.1	yes	yes	11	0.0
80	2	4	3	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.2	yes	yes	11	9.1
80	2	4	4	0.0	yes	yes	10	-	0.0	yes	yes	11	-	0.1	yes	yes	10	10.0
80	2	4	5	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.1	yes	yes	11	9.1
80	2	8	1	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7	0.1	yes	yes	6	16.7
80	2	8	2	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.1	yes	yes	6	16.7
80	2	8	3	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.1	yes	yes	6	16.7
80	2	8	4	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.1	yes	yes	6	16.7
80	2	8	5	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.2	yes	yes	6	16.7
80	3	1	1	0.1	yes	yes	28	-	0.1	yes	yes	28	-	0.4	yes	yes	28	0.0
80	3	1	2	0.0	yes	yes	27	-	0.1	yes	yes	28	-	0.4	yes	yes	27	3.7
80	3	1	3	0.1	yes	yes	28	-	0.1	yes	yes	28	3.6	0.3	yes	yes	28	3.6
80	3	1	4	0.1	yes	yes	28	-	0.0	yes	yes	28	-	0.5	yes	yes	28	0.0
80	3	1	5	0.0	yes	yes	27	-	0.0	yes	yes	28	0.0	0.3	yes	yes	27	3.7
80	3	2	1	0.0	yes	yes	15	-	0.0	yes	yes	14	-	0.3	yes	yes	15	0.0
80	3	2	2	0.0	yes	yes	15	0.0	0.0	yes	yes	15	0.0	0.2	yes	yes	15	0.0
80	3	2	3	0.0	yes	yes	15	-	0.0	yes	yes	15	-	0.2	yes	yes	15	0.0
80	3	2	4	0.0	yes	yes	15	-	0.0	yes	yes	15	-	0.2	yes	yes	15	0.0
80	3	2	5	0.0	yes	yes	14	-	0.0	yes	yes	14	7.1	0.2	yes	yes	14	0.0
80	3	4	1	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.2	yes	yes	8	12.5
80	3	4	2	0.0	yes	yes	8	12.5	0.0	yes	yes	8	-	0.2	yes	yes	8	12.5
80	3	4	3	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.2	yes	yes	8	12.5
80	3	4	4	0.0	yes	yes	7	-	0.0	yes	yes	8	-	0.3	yes	yes	7	14.3
80	3	4	5	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.1	yes	yes	8	12.5
80	3	8	1	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.2	yes	yes	5	0.0
80	3	8	2	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.2	yes	yes	5	0.0
80	3	8	3	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.2	yes	yes	5	0.0
80	3	8	4	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.2	yes	yes	5	20.0
80	3	8	5	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.1	yes	yes	5	0.0

Table A.9: Problem $V, Q|sd, u, r_i, d_i|CLT$ (part 7)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
160	1	1	1	0.3	yes	yes	158	0.0	0.3	yes	yes	160	1.3	0.6	yes	yes	158	0.0
160	1	1	2	0.3	yes	yes	161	-	0.3	yes	yes	160	0.6	0.5	yes	yes	161	0.0
160	1	1	3	0.3	yes	yes	161	-	0.4	yes	yes	161	-	0.6	yes	yes	161	0.0
160	1	1	4	0.2	yes	yes	160	-	0.4	yes	yes	160	-	0.7	yes	yes	160	1.9
160	1	1	5	0.2	yes	yes	161	-	0.2	yes	yes	162	-	0.6	yes	yes	161	0.6
160	1	2	1	0.1	yes	yes	80	-	0.1	yes	yes	80	-	0.4	yes	yes	80	0.0
160	1	2	2	0.1	yes	yes	81	-	0.1	yes	yes	81	-	0.4	yes	yes	81	1.2
160	1	2	3	0.0	yes	yes	80	-	0.1	yes	yes	81	0.0	0.4	yes	yes	80	0.0
160	1	2	4	0.1	yes	yes	81	-	0.1	yes	yes	82	-	0.6	yes	yes	81	2.5
160	1	2	5	0.1	yes	yes	81	-	0.1	yes	yes	80	-	0.4	yes	yes	81	0.0
160	1	4	1	0.0	yes	yes	41	-	0.0	yes	yes	41	2.4	0.3	yes	yes	41	2.4
160	1	4	2	0.0	yes	yes	41	-	0.1	yes	yes	41	-	0.4	yes	yes	41	0.0
160	1	4	3	0.0	yes	yes	40	-	0.1	yes	yes	41	0.0	0.3	yes	yes	40	0.0
160	1	4	4	0.0	yes	yes	41	-	0.0	yes	yes	41	2.4	0.3	yes	yes	41	2.4
160	1	4	5	0.0	yes	yes	41	-	0.1	yes	yes	41	-	0.3	yes	yes	41	0.0
160	1	8	1	0.0	yes	yes	21	-	0.0	yes	yes	21	-	0.3	yes	yes	21	4.8
160	1	8	2	0.0	yes	yes	20	-	0.0	yes	yes	21	-	0.2	yes	yes	20	5.0
160	1	8	3	0.0	yes	yes	21	-	0.0	yes	yes	21	-	0.3	yes	yes	21	4.8
160	1	8	4	0.0	yes	yes	21	-	0.0	yes	yes	21	-	0.2	yes	yes	21	0.0
160	1	8	5	0.0	yes	yes	21	-	0.0	yes	yes	21	4.8	0.2	yes	yes	21	4.8
160	2	1	1	0.3	yes	yes	81	-	0.2	yes	yes	80	-	1.3	yes	yes	81	1.2
160	2	1	2	0.1	yes	yes	81	-	0.3	yes	yes	80	-	1.1	yes	yes	81	1.2
160	2	1	3	0.2	yes	yes	81	-	0.2	yes	yes	81	-	1.0	yes	yes	81	0.0
160	2	1	4	0.2	yes	yes	80	-	0.4	yes	yes	79	1.3	1.3	yes	yes	80	1.3
160	2	1	5	0.3	yes	yes	82	-	0.3	yes	yes	81	1.2	1.1	yes	yes	82	0.0
160	2	2	1	0.1	yes	yes	41	0.0	0.1	yes	yes	41	-	0.7	yes	yes	41	0.0
160	2	2	2	0.1	yes	yes	41	-	0.1	yes	yes	41	0.0	0.6	yes	yes	41	0.0
160	2	2	3	0.1	yes	yes	41	-	0.1	yes	yes	40	-	0.7	yes	yes	41	2.4
160	2	2	4	0.1	yes	yes	41	-	0.1	yes	yes	41	-	0.9	yes	yes	41	2.4
160	2	2	5	0.1	yes	yes	41	-	0.1	yes	yes	41	-	0.9	yes	yes	41	0.0
160	2	4	1	0.1	yes	yes	21	4.8	0.1	yes	yes	21	-	0.8	yes	yes	21	4.8
160	2	4	2	0.0	yes	yes	21	-	0.0	yes	yes	21	-	0.8	yes	yes	21	4.8
160	2	4	3	0.0	yes	yes	21	-	0.0	yes	yes	20	-	0.8	yes	yes	21	0.0
160	2	4	4	0.0	yes	yes	21	-	0.0	yes	yes	20	-	0.8	yes	yes	21	0.0
160	2	4	5	0.0	yes	yes	21	-	0.1	yes	yes	20	-	0.8	yes	yes	21	4.8
160	2	8	1	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.7	yes	yes	11	9.1
160	2	8	2	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.7	yes	yes	11	0.0
160	2	8	3	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.7	yes	yes	11	9.1
160	2	8	4	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.6	yes	yes	11	9.1
160	2	8	5	0.0	yes	yes	11	-	0.0	yes	yes	11	9.1	0.7	yes	yes	11	9.1
160	3	1	1	0.3	yes	yes	55	-	0.3	yes	yes	54	3.7	1.3	yes	yes	55	0.0
160	3	1	2	0.2	yes	yes	54	-	0.3	yes	yes	54	-	1.3	yes	yes	54	0.0
160	3	1	3	0.1	yes	yes	54	-	0.3	yes	yes	53	0.0	1.2	yes	yes	54	0.0
160	3	1	4	0.2	yes	yes	55	-	0.3	yes	yes	55	-	1.8	yes	yes	55	0.0
160	3	1	5	0.2	yes	yes	55	-	0.2	yes	yes	54	-	2.0	yes	yes	55	1.8
160	3	2	1	0.1	yes	yes	28	-	0.1	yes	yes	28	3.6	1.3	yes	yes	28	0.0
160	3	2	2	0.1	yes	yes	28	-	0.1	yes	yes	28	-	1.3	yes	yes	28	0.0
160	3	2	3	0.1	yes	yes	27	-	0.1	yes	yes	28	-	1.3	yes	yes	27	3.7
160	3	2	4	0.1	yes	yes	28	-	0.1	yes	yes	28	0.0	1.3	yes	yes	28	3.6
160	3	2	5	0.1	yes	yes	28	-	0.1	yes	yes	28	-	1.3	yes	yes	28	0.0
160	3	4	1	0.0	yes	yes	15	-	0.1	yes	yes	15	-	1.1	yes	yes	15	0.0
160	3	4	2	0.0	yes	yes	15	-	0.1	yes	yes	15	-	1.1	yes	yes	15	0.0
160	3	4	3	0.0	yes	yes	15	-	0.1	yes	yes	15	-	1.1	yes	yes	15	0.0
160	3	4	4	0.0	yes	yes	14	-	0.1	yes	yes	15	-	1.1	yes	yes	14	7.1
160	3	4	5	0.0	yes	yes	15	-	0.1	yes	yes	15	-	1.1	yes	yes	15	0.0
160	3	8	1	0.0	yes	yes	8	-	0.0	yes	yes	8	-	1.1	yes	yes	8	12.5
160	3	8	2	0.0	yes	yes	8	-	0.0	yes	yes	8	-	1.1	yes	yes	8	0.0
160	3	8	3	0.0	yes	yes	8	-	0.0	yes	yes	8	-	1.0	yes	yes	8	0.0
160	3	8	4	0.0	yes	yes	8	-	0.0	yes	yes	8	-	1.0	yes	yes	8	0.0
160	3	8	5	0.0	yes	yes	8	-	0.0	yes	yes	8	-	1.0	yes	yes	8	12.5

Table A.10: Problem $V, Q|sd, u, r_i, d_i|CLT$ (part 8)

n	V	Q	instance	CPU(s)	CLT	opt	gapGr(%)
20	1	1	1	0.1	13	yes	7.7
20	1	1	2	0.1	15	yes	0.0
20	1	1	3	0.1	12	yes	0.0
20	1	1	4	0.1	13	yes	0.0
20	1	1	5	0.0	12	yes	0.0
20	1	2	1	0.1	10	yes	10.0
20	1	2	2	0.1	11	yes	0.0
20	1	2	3	0.0	9	yes	11.1
20	1	2	4	0.1	10	yes	20.0
20	1	2	5	0.0	9	yes	0.0
20	1	4	1	0.1	10	yes	10.0
20	1	4	2	0.1	10	yes	10.0
20	1	4	3	0.1	9	yes	11.1
20	1	4	4	0.1	8	yes	12.5
20	1	4	5	0.0	7	yes	14.3
20	1	8	1	0.0	8	yes	12.5
20	1	8	2	0.1	9	yes	22.2
20	1	8	3	0.0	8	yes	12.5
20	1	8	4	0.1	8	yes	25.0
20	1	8	5	0.0	8	yes	12.5
20	2	1	1	0.1	7	yes	14.3
20	2	1	2	0.1	8	yes	0.0
20	2	1	3	0.1	6	yes	0.0
20	2	1	4	0.1	7	yes	0.0
20	2	1	5	0.0	6	yes	0.0
20	2	2	1	0.1	5	yes	20.0
20	2	2	2	0.1	7	yes	14.3
20	2	2	3	0.1	5	yes	0.0
20	2	2	4	0.1	5	yes	20.0
20	2	2	5	0.1	5	yes	0.0
20	2	4	1	0.1	5	yes	20.0
20	2	4	2	0.1	6	yes	0.0
20	2	4	3	0.1	4	yes	25.0
20	2	4	4	0.1	5	yes	20.0
20	2	4	5	0.0	6	yes	0.0
20	2	8	1	0.0	5	yes	20.0
20	2	8	2	0.1	5	yes	20.0
20	2	8	3	0.1	5	yes	0.0
20	2	8	4	0.1	4	yes	25.0
20	2	8	5	0.1	5	yes	0.0
20	3	1	1	0.1	5	yes	20.0
20	3	1	2	0.1	6	yes	0.0
20	3	1	3	0.1	4	yes	25.0
20	3	1	4	0.1	5	yes	0.0
20	3	1	5	0.1	4	yes	0.0
20	3	2	1	0.1	4	yes	25.0
20	3	2	2	0.1	5	yes	0.0
20	3	2	3	0.1	3	yes	33.3
20	3	2	4	0.1	4	yes	25.0
20	3	2	5	0.1	3	yes	33.3
20	3	4	1	0.1	4	yes	0.0
20	3	4	2	0.1	5	yes	0.0
20	3	4	3	0.1	3	yes	33.3
20	3	4	4	0.1	3	yes	66.7
20	3	4	5	0.1	3	yes	33.3
20	3	8	1	0.1	3	yes	33.3
20	3	8	2	0.1	4	yes	25.0
20	3	8	3	0.0	2	yes	50.0
20	3	8	4	0.1	4	yes	25.0
20	3	8	5	0.0	3	yes	0.0

n	V	Q	instance	CPU(s)	CLT	opt	gapGr(%)
40	1	1	1	0.5	21	yes	4.8
40	1	1	2	0.3	25	yes	0.0
40	1	1	3	0.2	29	yes	3.4
40	1	1	4	0.2	27	yes	0.0
40	1	1	5	0.2	25	yes	0.0
40	1	2	1	0.2	18	yes	0.0
40	1	2	2	0.2	18	yes	5.6
40	1	2	3	0.2	20	yes	0.0
40	1	2	4	0.1	21	yes	4.8
40	1	2	5	0.3	18	yes	5.6
40	1	4	1	4.4	13	yes	23.1
40	1	4	2	0.3	16	yes	18.8
40	1	4	3	0.3	16	yes	12.5
40	1	4	4	0.2	19	yes	5.3
40	1	4	5	0.7	21	yes	4.8
40	1	8	1	0.4	13	yes	15.4
40	1	8	2	0.5	16	yes	6.3
40	1	8	3	0.9	17	yes	11.8
40	1	8	4	0.5	15	yes	13.3
40	1	8	5	0.4	12	yes	8.3
40	2	1	1	0.6	11	yes	0.0
40	2	1	2	0.6	13	yes	0.0
40	2	1	3	0.3	15	yes	0.0
40	2	1	4	0.2	14	yes	0.0
40	2	1	5	0.3	13	yes	7.7
40	2	2	1	0.3	10	yes	10.0
40	2	2	2	0.2	9	yes	22.2
40	2	2	3	0.3	10	yes	10.0
40	2	2	4	0.2	10	yes	10.0
40	2	2	5	0.4	10	yes	0.0
40	2	4	1	0.3	7	yes	14.3
40	2	4	2	0.2	8	yes	0.0
40	2	4	3	0.9	11	yes	9.1
40	2	4	4	0.2	9	yes	11.1
40	2	4	5	0.2	8	yes	0.0
40	2	8	1	1.2	9	yes	11.1
40	2	8	2	1.5	9	yes	11.1
40	2	8	3	0.4	9	yes	11.1
40	2	8	4	0.2	9	yes	11.1
40	2	8	5	0.2	8	yes	12.5
40	3	1	1	0.6	8	yes	0.0
40	3	1	2	0.4	9	yes	0.0
40	3	1	3	0.4	10	yes	10.0
40	3	1	4	0.3	9	yes	11.1
40	3	1	5	0.4	9	yes	0.0
40	3	2	1	0.6	6	yes	16.7
40	3	2	2	0.4	7	yes	14.3
40	3	2	3	0.4	8	yes	12.5
40	3	2	4	0.3	8	yes	12.5
40	3	2	5	0.2	7	yes	14.3
40	3	4	1	0.2	6	yes	16.7
40	3	4	2	0.3	5	yes	20.0
40	3	4	3	0.8	8	yes	0.0
40	3	4	4	0.2	6	yes	16.7
40	3	4	5	0.3	5	yes	20.0
40	3	8	1	0.2	5	yes	20.0
40	3	8	2	0.2	4	yes	25.0
40	3	8	3	0.3	6	yes	16.7
40	3	8	4	0.3	5	yes	20.0
40	3	8	5	0.2	5	yes	20.0

Table A.11: Problem $V, Q|sd|CLT$ (part 1)

n	V	Q	instance	CPU(s)	CLT	opt	gapGr(%)
80	1	1	1	2.5	45	yes	2.2
80	1	1	2	3.3	44	yes	0.0
80	1	1	3	3.2	48	yes	2.1
80	1	1	4	2.3	49	yes	2.0
80	1	1	5	1.0	46	yes	2.2
80	1	2	1	2.5	36	yes	2.8
80	1	2	2	4.7	35	yes	5.7
80	1	2	3	2.6	35	yes	2.9
80	1	2	4	2.5	38	yes	2.6
80	1	2	5	2.1	34	yes	0.0
80	1	4	1	20.0	29	yes	3.4
80	1	4	2	20.8	28	yes	7.1
80	1	4	3	15.0	28	yes	17.9
80	1	4	4	15.0	35	yes	11.4
80	1	4	5	8.7	32	yes	9.4
80	1	8	1	23.4	25	yes	20.0
80	1	8	2	27.7	26	yes	15.4
80	1	8	3	14.3	29	yes	13.8
80	1	8	4	19.9	31	yes	6.5
80	1	8	5	29.5	27	yes	3.7
80	2	1	1	2.8	23	yes	0.0
80	2	1	2	2.9	22	yes	0.0
80	2	1	3	2.7	24	yes	4.2
80	2	1	4	2.4	25	yes	4.0
80	2	1	5	1.7	23	yes	0.0
80	2	2	1	5.2	17	yes	11.8
80	2	2	2	7.5	17	yes	5.9
80	2	2	3	3.2	19	yes	5.3
80	2	2	4	1.4	19	yes	5.3
80	2	2	5	2.3	19	yes	0.0
80	2	4	1	4.1	15	yes	6.7
80	2	4	2	4.0	14	yes	7.1
80	2	4	3	9.6	15	yes	13.3
80	2	4	4	6.2	18	yes	5.6
80	2	4	5	3.0	16	yes	12.5
80	2	8	1	2.2	13	yes	7.7
80	2	8	2	9.6	15	yes	6.7
80	2	8	3	6.7	13	yes	23.1
80	2	8	4	91.1	15	yes	13.3
80	2	8	5	12.1	15	yes	13.3
80	3	1	1	7.0	15	yes	6.7
80	3	1	2	4.9	15	yes	0.0
80	3	1	3	3.1	16	yes	6.3
80	3	1	4	2.7	17	yes	5.9
80	3	1	5	1.8	16	yes	0.0
80	3	2	1	5.5	13	yes	7.7
80	3	2	2	8.0	11	yes	9.1
80	3	2	3	6.6	11	yes	18.2
80	3	2	4	5.4	12	yes	0.0
80	3	2	5	3.4	12	yes	0.0
80	3	4	1	1.9	10	yes	10.0
80	3	4	2	12.0	10	yes	10.0
80	3	4	3	4.1	10	yes	20.0
80	3	4	4	8.7	11	yes	9.1
80	3	4	5	3.4	11	yes	18.2
80	3	8	1	7.8	10	yes	20.0
80	3	8	2	5.9	8	yes	25.0
80	3	8	3	6.0	11	yes	18.2
80	3	8	4	8.6	10	yes	20.0
80	3	8	5	73.0	10	yes	0.0

n	V	Q	instance	CPU(s)	CLT	opt	gapGr(%)
160	1	1	1	19.3	89	yes	1.1
160	1	1	2	14.3	91	yes	1.1
160	1	1	3	22.6	83	yes	0.0
160	1	1	4	14.2	87	yes	1.1
160	1	1	5	17.5	87	yes	0.0
160	1	2	1	422.5	70	yes	0.0
160	1	2	2	12.7	69	yes	1.4
160	1	2	3	56.8	62	yes	1.6
160	1	2	4	15.3	63	yes	1.6
160	1	2	5	7.3	63	yes	1.6
160	1	4	1	415.7	56	yes	10.7
160	1	4	2	59.9	57	yes	10.5
160	1	4	3	300.5	52	yes	15.4
160	1	4	4	310.6	56	yes	10.7
160	1	4	5	51.6	53	yes	5.7
160	1	8	1	491.5	57	yes	12.3
160	1	8	2	587.8	50	yes	16.0
160	1	8	3	76.3	45	yes	13.3
160	1	8	4	75.0	51	yes	11.8
160	1	8	5	230.9	55	yes	9.1
160	2	1	1	39.6	45	yes	0.0
160	2	1	2	23.0	46	yes	2.2
160	2	1	3	31.7	42	yes	2.4
160	2	1	4	41.4	44	yes	2.3
160	2	1	5	18.5	44	yes	2.3
160	2	2	1	19.0	34	yes	2.9
160	2	2	2	22.9	34	yes	2.9
160	2	2	3	31.0	32	yes	3.1
160	2	2	4	45.9	33	yes	3.0
160	2	2	5	29.9	33	yes	3.0
160	2	4	1	92.9	28	yes	14.3
160	2	4	2	322.4	29	yes	6.9
160	2	4	3	320.0	26	yes	15.4
160	2	4	4	198.6	26	yes	11.5
160	2	4	5	131.7	28	yes	10.7
160	2	8	1	1800.1	31	yes	9.7
160	2	8	2	127.6	24	yes	8.3
160	2	8	3	160.4	25	yes	12.0
160	2	8	4	97.7	26	yes	15.4
160	2	8	5	82.9	21	yes	9.5
160	3	1	1	33.6	30	yes	3.3
160	3	1	2	48.2	31	yes	0.0
160	3	1	3	45.0	28	yes	0.0
160	3	1	4	44.5	29	yes	3.4
160	3	1	5	28.3	29	yes	3.4
160	3	2	1	97.5	23	yes	0.0
160	3	2	2	21.5	24	yes	4.2
160	3	2	3	76.5	22	yes	4.5
160	3	2	4	27.4	21	yes	4.8
160	3	2	5	28.4	21	yes	4.8
160	3	4	1	111.9	21	yes	14.3
160	3	4	2	48.5	21	yes	4.8
160	3	4	3	147.5	19	yes	10.5
160	3	4	4	70.8	19	yes	10.5
160	3	4	5	53.2	19	yes	10.5
160	3	8	1	172.3	17	yes	11.8
160	3	8	2	180.4	20	yes	10.0
160	3	8	3	226.3	19	yes	10.5
160	3	8	4	74.1	17	yes	17.6
160	3	8	5	160.4	19	yes	10.5

Table A.12: Problem $V, Q|sd|CLT$ (part 2)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
20	1	1	1	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	13	7.7
20	1	1	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	17	5.9
20	1	1	3	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	12	8.3
20	1	1	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	14	7.1
20	1	1	5	-	no	-	-	-	0.0	yes	yes	13	-	0.0	yes	yes	12	8.3
20	1	2	1	-	no	-	-	-	0.0	yes	yes	11	-	0.0	yes	yes	12	8.3
20	1	2	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	11	9.1
20	1	2	3	-	no	-	-	-	0.0	yes	yes	12	-	0.0	yes	yes	11	9.1
20	1	2	4	-	no	-	-	-	0.0	yes	yes	12	-	0.0	yes	yes	12	8.3
20	1	2	5	-	no	-	-	-	0.0	yes	yes	10	-	0.0	yes	yes	9	33.3
20	1	4	1	-	no	-	-	-	0.0	yes	yes	11	-	0.0	yes	yes	12	16.7
20	1	4	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	12	16.7
20	1	4	3	-	no	-	-	-	0.0	yes	yes	9	-	0.0	yes	yes	10	30.0
20	1	4	4	-	no	-	-	-	0.0	yes	yes	11	-	0.0	yes	yes	10	10.0
20	1	4	5	0.0	yes	yes	9	11.1	0.0	yes	yes	9	-	0.0	yes	yes	9	11.1
20	1	8	1	0.0	yes	yes	9	-	0.0	yes	yes	10	-	0.0	yes	yes	9	33.3
20	1	8	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	10	0.0
20	1	8	3	-	no	-	-	-	0.0	yes	yes	10	-	0.0	yes	yes	10	30.0
20	1	8	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	12	8.3
20	1	8	5	0.0	yes	yes	9	-	0.0	yes	yes	10	20.0	0.0	yes	yes	9	11.1
20	2	1	1	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	7	14.3
20	2	1	2	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	8	25.0
20	2	1	3	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	6	33.3
20	2	1	4	-	no	-	-	-	0.0	yes	yes	8	-	0.0	yes	yes	8	0.0
20	2	1	5	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	6	16.7
20	2	2	1	-	no	-	-	-	0.0	yes	yes	7	-	0.0	yes	yes	5	20.0
20	2	2	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	7	14.3
20	2	2	3	-	no	-	-	-	0.0	yes	yes	6	-	0.0	yes	yes	5	20.0
20	2	2	4	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	33.3
20	2	2	5	-	no	-	-	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
20	2	4	1	0.0	yes	yes	6	-	-	no	-	-	-	0.0	yes	yes	5	40.0
20	2	4	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	6	16.7
20	2	4	3	0.0	yes	yes	6	-	0.0	yes	yes	5	-	0.0	yes	yes	6	16.7
20	2	4	4	-	no	-	-	-	0.0	yes	yes	6	-	0.0	yes	yes	6	33.3
20	2	4	5	-	no	-	-	-	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
20	2	8	1	0.0	yes	yes	6	-	0.0	yes	yes	5	-	0.0	yes	yes	5	40.0
20	2	8	2	-	no	-	-	-	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
20	2	8	3	0.0	yes	yes	6	-	0.0	yes	yes	5	-	0.0	yes	yes	6	16.7
20	2	8	4	0.0	yes	yes	5	-	0.0	yes	yes	4	-	0.0	yes	yes	5	40.0
20	2	8	5	0.0	yes	yes	6	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
20	3	1	1	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	5	20.0
20	3	1	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	6	0.0
20	3	1	3	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	4	50.0
20	3	1	4	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	6	16.7
20	3	1	5	-	no	-	-	-	0.0	yes	yes	4	-	0.0	yes	yes	5	0.0
20	3	2	1	-	no	-	-	-	0.0	yes	yes	5	-	0.0	yes	yes	4	25.0
20	3	2	2	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	0.0
20	3	2	3	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	3	2	4	-	no	-	-	-	0.0	yes	yes	5	-	0.0	yes	yes	5	0.0
20	3	2	5	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	3	4	1	-	no	-	-	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	3	4	2	-	no	-	-	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
20	3	4	3	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	3	4	4	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	3	4	5	-	no	-	-	-	0.0	yes	yes	4	-	0.0	yes	yes	3	66.7
20	3	8	1	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	3	8	2	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	4	0.0
20	3	8	3	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0	0.0	yes	yes	4	0.0
20	3	8	4	-	no	-	-	-	0.0	yes	yes	4	-	0.0	yes	yes	5	20.0
20	3	8	5	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	0.0

Table A.13: Problem $V, Q|sd, r_i, d_i|CLT$ (part 1)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
40	1	1	1	-	no	-	-	-	0.0	yes	yes	24	-	0.1	yes	yes	24	16.7
40	1	1	2	-	no	-	-	-	-	no	-	-	-	0.2	yes	yes	25	4.0
40	1	1	3	-	no	-	-	-	-	no	-	-	-	0.2	yes	yes	29	0.0
40	1	1	4	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	28	3.6
40	1	1	5	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	28	0.0
40	1	2	1	-	no	-	-	-	-	no	-	-	-	0.2	yes	yes	19	15.8
40	1	2	2	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	20	15.0
40	1	2	3	0.1	yes	yes	21	-	0.1	yes	yes	21	-	0.1	yes	yes	21	0.0
40	1	2	4	-	no	-	-	-	0.1	yes	yes	21	-	0.1	yes	yes	22	0.0
40	1	2	5	-	no	-	-	-	0.1	yes	yes	20	-	0.3	yes	yes	19	15.8
40	1	4	1	0.0	yes	yes	19	-	0.0	yes	yes	18	-	0.1	yes	yes	18	11.1
40	1	4	2	0.0	yes	yes	19	-	-	no	-	-	-	0.1	yes	yes	18	11.1
40	1	4	3	0.1	yes	yes	18	-	0.1	yes	yes	19	-	0.3	yes	yes	17	11.8
40	1	4	4	-	no	-	-	-	0.2	yes	yes	20	-	0.1	yes	yes	21	19.0
40	1	4	5	-	no	-	-	-	-	no	-	-	-	0.3	yes	yes	22	9.1
40	1	8	1	0.0	yes	yes	19	-	0.0	yes	yes	17	-	0.1	yes	yes	19	5.3
40	1	8	2	0.0	yes	yes	18	-	0.1	yes	yes	19	-	0.1	yes	yes	17	29.4
40	1	8	3	-	no	-	-	-	-	no	-	-	-	0.3	yes	yes	20	20.0
40	1	8	4	0.0	yes	yes	17	-	0.1	yes	yes	18	-	0.1	yes	yes	17	17.6
40	1	8	5	0.0	yes	yes	17	-	0.0	yes	yes	17	-	0.1	yes	yes	17	0.0
40	2	1	1	-	no	-	-	-	0.1	yes	yes	13	-	0.3	yes	yes	12	16.7
40	2	1	2	-	no	-	-	-	0.0	yes	yes	13	-	0.2	yes	yes	14	7.1
40	2	1	3	-	no	-	-	-	-	no	-	-	-	0.4	yes	yes	15	6.7
40	2	1	4	-	no	-	-	-	-	no	-	-	-	0.2	yes	yes	14	0.0
40	2	1	5	-	no	-	-	-	-	no	-	-	-	0.2	yes	yes	13	7.7
40	2	2	1	-	no	-	-	-	0.1	yes	yes	10	-	0.2	yes	yes	10	20.0
40	2	2	2	-	no	-	-	-	0.1	yes	yes	10	-	0.2	yes	yes	10	10.0
40	2	2	3	-	no	-	-	-	0.1	yes	yes	10	-	0.2	yes	yes	11	9.1
40	2	2	4	0.0	yes	yes	10	-	0.0	yes	yes	10	-	0.1	yes	yes	10	20.0
40	2	2	5	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	10	10.0
40	2	4	1	0.0	yes	yes	9	-	0.2	yes	yes	10	-	0.1	yes	yes	8	37.5
40	2	4	2	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.1	yes	yes	9	11.1
40	2	4	3	-	no	-	-	-	0.1	yes	yes	11	-	0.5	yes	yes	11	0.0
40	2	4	4	0.0	yes	yes	10	-	0.0	yes	yes	10	-	0.2	yes	yes	10	10.0
40	2	4	5	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.1	yes	yes	9	11.1
40	2	8	1	0.2	yes	yes	10	-	0.1	yes	yes	10	-	0.3	yes	yes	9	33.3
40	2	8	2	0.1	yes	yes	11	-	0.0	yes	yes	10	-	0.3	yes	yes	11	0.0
40	2	8	3	0.0	yes	yes	11	-	0.1	yes	yes	11	-	0.2	yes	yes	11	9.1
40	2	8	4	-	no	-	-	-	0.0	yes	yes	10	-	0.1	yes	yes	11	9.1
40	2	8	5	0.0	yes	yes	10	-	0.0	yes	yes	9	-	0.1	yes	yes	10	10.0
40	3	1	1	-	no	-	-	-	0.0	yes	yes	9	-	0.3	yes	yes	8	25.0
40	3	1	2	-	no	-	-	-	0.0	yes	yes	9	-	0.3	yes	yes	9	11.1
40	3	1	3	-	no	-	-	-	-	no	-	-	-	0.3	yes	yes	10	10.0
40	3	1	4	-	no	-	-	-	0.0	yes	yes	9	-	0.2	yes	yes	10	20.0
40	3	1	5	-	no	-	-	-	0.0	yes	yes	9	-	0.3	yes	yes	9	0.0
40	3	2	1	0.0	yes	yes	7	-	0.0	yes	yes	7	-	0.2	yes	yes	7	14.3
40	3	2	2	-	no	-	-	-	0.1	yes	yes	8	-	0.2	yes	yes	7	14.3
40	3	2	3	-	no	-	-	-	0.1	yes	yes	9	-	0.4	yes	yes	8	12.5
40	3	2	4	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.2	yes	yes	8	12.5
40	3	2	5	0.0	yes	yes	7	-	-	no	-	-	-	0.2	yes	yes	7	28.6
40	3	4	1	0.0	yes	yes	6	16.7	0.0	yes	yes	7	-	0.1	yes	yes	6	16.7
40	3	4	2	0.0	yes	yes	7	-	0.0	yes	yes	6	-	0.1	yes	yes	6	16.7
40	3	4	3	-	no	-	-	-	-	no	-	-	-	0.2	yes	yes	8	12.5
40	3	4	4	0.0	yes	yes	7	-	0.0	yes	yes	7	-	0.1	yes	yes	7	14.3
40	3	4	5	0.0	yes	yes	7	-	0.0	yes	yes	8	-	0.1	yes	yes	7	14.3
40	3	8	1	0.0	yes	yes	7	-	0.0	yes	yes	7	-	0.1	yes	yes	7	14.3
40	3	8	2	0.0	yes	yes	7	-	0.0	yes	yes	6	-	0.1	yes	yes	7	14.3
40	3	8	3	-	no	-	-	-	0.1	yes	yes	7	-	0.1	yes	yes	6	50.0
40	3	8	4	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.1	yes	yes	6	16.7
40	3	8	5	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.1	yes	yes	6	16.7

Table A.14: Problem $V, Q|sd, r_i, d_i|CLT$ (part 2)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
80	1	1	1	-	no	-	-	-	0.4	yes	yes	45	-	1.8	yes	yes	47	4.3
80	1	1	2	-	no	-	-	-	0.2	yes	yes	48	-	0.8	yes	yes	47	2.1
80	1	1	3	-	no	-	-	-	0.2	yes	yes	50	-	1.7	yes	yes	49	4.1
80	1	1	4	-	no	-	-	-	-	no	-	-	-	1.5	yes	yes	49	6.1
80	1	1	5	-	no	-	-	-	0.2	yes	yes	47	-	0.8	yes	yes	47	4.3
80	1	2	1	0.8	yes	yes	38	-	1.0	yes	yes	36	-	1.5	yes	yes	38	7.9
80	1	2	2	0.7	yes	yes	42	-	1.4	yes	yes	36	-	3.4	yes	yes	41	9.8
80	1	2	3	-	no	-	-	-	2.5	yes	yes	38	-	3.9	yes	yes	36	13.9
80	1	2	4	-	no	-	-	-	0.8	yes	yes	39	-	0.6	yes	yes	38	10.5
80	1	2	5	0.2	yes	yes	38	-	0.3	yes	yes	37	-	1.7	yes	yes	38	7.9
80	1	4	1	0.5	yes	yes	34	-	1.1	yes	yes	35	-	1.7	yes	yes	33	12.1
80	1	4	2	0.2	yes	yes	34	-	1.3	yes	yes	35	11.4	1.3	yes	yes	34	11.8
80	1	4	3	0.2	yes	yes	35	-	0.2	yes	yes	34	-	1.5	yes	yes	34	17.6
80	1	4	4	-	no	-	-	-	1.9	yes	yes	38	-	7.1	yes	yes	37	8.1
80	1	4	5	-	no	-	-	-	56.6	yes	yes	34	-	2.0	yes	yes	35	8.6
80	1	8	1	0.1	yes	yes	31	-	0.4	yes	yes	33	-	0.2	yes	yes	31	9.7
80	1	8	2	0.2	yes	yes	33	-	1.1	yes	yes	34	-	0.3	yes	yes	33	9.1
80	1	8	3	0.3	yes	yes	33	-	0.3	yes	yes	33	-	2.4	yes	yes	32	12.5
80	1	8	4	1.4	yes	yes	35	-	2.1	yes	yes	34	-	8.0	yes	yes	34	17.6
80	1	8	5	0.1	yes	yes	31	-	0.2	yes	yes	34	-	0.2	yes	yes	31	9.7
80	2	1	1	-	no	-	-	-	-	no	-	-	-	1.6	yes	yes	24	4.2
80	2	1	2	-	no	-	-	-	-	no	-	-	-	1.9	yes	yes	23	4.3
80	2	1	3	-	no	-	-	-	0.2	yes	yes	25	-	3.0	yes	yes	24	4.2
80	2	1	4	-	no	-	-	-	0.3	yes	yes	25	-	2.1	yes	yes	25	4.0
80	2	1	5	-	no	-	-	-	0.4	yes	yes	23	-	1.2	yes	yes	24	4.2
80	2	2	1	0.7	yes	yes	18	-	1.1	yes	yes	18	-	2.3	yes	yes	18	16.7
80	2	2	2	0.4	yes	yes	19	-	0.7	yes	yes	18	-	1.4	yes	yes	19	10.5
80	2	2	3	0.9	yes	yes	21	-	7.2	yes	yes	19	-	4.4	yes	yes	19	21.1
80	2	2	4	0.6	yes	yes	20	-	0.5	yes	yes	20	-	3.4	yes	yes	19	10.5
80	2	2	5	0.2	yes	yes	19	-	0.8	yes	yes	19	-	0.8	yes	yes	19	15.8
80	2	4	1	0.2	yes	yes	16	-	1.8	yes	yes	18	-	1.4	yes	yes	16	6.3
80	2	4	2	0.2	yes	yes	16	-	0.2	yes	yes	16	-	0.4	yes	yes	16	12.5
80	2	4	3	0.2	yes	yes	19	-	0.3	yes	yes	18	-	2.0	yes	yes	18	16.7
80	2	4	4	36.1	yes	yes	18	-	3.6	yes	yes	18	-	41.1	yes	yes	18	11.1
80	2	4	5	0.2	yes	yes	17	-	0.2	yes	yes	18	-	0.8	yes	yes	17	11.8
80	2	8	1	0.2	yes	yes	17	-	0.4	yes	yes	18	-	0.5	yes	yes	17	5.9
80	2	8	2	0.3	yes	yes	18	-	0.8	yes	yes	18	-	1.1	yes	yes	18	11.1
80	2	8	3	0.2	yes	yes	17	-	0.2	yes	yes	17	-	5.8	yes	yes	17	5.9
80	2	8	4	0.2	yes	yes	17	-	0.5	yes	yes	16	-	0.9	yes	yes	17	5.9
80	2	8	5	0.1	yes	yes	17	-	2.3	yes	yes	17	-	0.5	yes	yes	17	5.9
80	3	1	1	-	no	-	-	-	0.3	yes	yes	16	-	3.0	yes	yes	16	6.3
80	3	1	2	-	no	-	-	-	0.5	yes	yes	16	-	2.8	yes	yes	15	6.7
80	3	1	3	-	no	-	-	-	-	no	-	-	-	2.7	yes	yes	17	5.9
80	3	1	4	-	no	-	-	-	0.2	yes	yes	17	-	2.2	yes	yes	17	0.0
80	3	1	5	-	no	-	-	-	0.3	yes	yes	16	-	1.5	yes	yes	16	6.3
80	3	2	1	4.3	yes	yes	13	-	0.6	yes	yes	13	-	1.7	yes	yes	13	7.7
80	3	2	2	0.2	yes	yes	14	-	0.3	yes	yes	13	-	1.8	yes	yes	14	7.1
80	3	2	3	0.2	yes	yes	13	-	0.7	yes	yes	12	-	1.0	yes	yes	13	15.4
80	3	2	4	0.3	yes	yes	13	-	0.3	yes	yes	12	-	1.6	yes	yes	13	7.7
80	3	2	5	0.1	yes	yes	13	-	0.2	yes	yes	12	-	1.0	yes	yes	13	7.7
80	3	4	1	0.5	yes	yes	12	-	0.1	yes	yes	11	-	3.2	yes	yes	12	8.3
80	3	4	2	0.1	yes	yes	12	-	0.3	yes	yes	12	-	0.8	yes	yes	12	16.7
80	3	4	3	0.1	yes	yes	12	-	6.6	yes	yes	12	-	0.7	yes	yes	12	8.3
80	3	4	4	0.2	yes	yes	12	-	0.8	yes	yes	12	-	0.3	yes	yes	12	8.3
80	3	4	5	0.3	yes	yes	12	-	0.8	yes	yes	13	-	1.3	yes	yes	12	25.0
80	3	8	1	0.3	yes	yes	12	-	0.4	yes	yes	12	-	2.5	yes	yes	12	8.3
80	3	8	2	0.1	yes	yes	11	9.1	0.1	yes	yes	11	-	0.5	yes	yes	11	9.1
80	3	8	3	0.4	yes	yes	13	-	0.9	yes	yes	12	-	3.7	yes	yes	13	15.4
80	3	8	4	1.3	yes	yes	11	-	1.3	yes	yes	11	-	0.7	yes	yes	11	27.3
80	3	8	5	0.2	yes	yes	13	-	0.3	yes	yes	12	-	0.6	yes	yes	12	16.7

Table A.15: Problem $V, Q|sd, r_i, d_i|CLT$ (part 3)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
160	1	1	1	-	no	-	-	-	15.7	yes	yes	91	-	87.4	yes	yes	89	5.6
160	1	1	2	-	no	-	-	-	2.4	yes	yes	92	-	35.3	yes	yes	94	5.3
160	1	1	3	-	no	-	-	-	9.5	yes	yes	84	-	49.1	yes	yes	84	3.6
160	1	1	4	-	no	-	-	-	8.4	yes	yes	88	-	46.8	yes	yes	88	3.4
160	1	1	5	-	no	-	-	-	3.1	yes	yes	87	-	9.5	yes	yes	88	1.1
160	1	2	1	339.4	yes	yes	76	-	107.9	yes	yes	73	-	1550.5	yes	yes	75	5.3
160	1	2	2	33.6	yes	yes	72	-	23.9	yes	yes	72	-	61.3	yes	yes	72	4.2
160	1	2	3	500.9	yes	yes	65	-	16.8	yes	yes	69	-	49.3	yes	yes	65	6.2
160	1	2	4	9.3	yes	yes	70	-	27.6	yes	yes	68	-	51.3	yes	yes	70	5.7
160	1	2	5	14.5	yes	yes	64	-	16.5	yes	yes	64	-	19.6	yes	yes	64	10.9
160	1	4	1	7.6	yes	yes	66	-	28.7	yes	yes	62	-	52.9	yes	yes	66	6.1
160	1	4	2	22.0	yes	yes	65	-	18.9	yes	yes	63	-	43.6	yes	yes	65	9.2
160	1	4	3	6.7	yes	yes	65	-	10.4	yes	yes	61	-	7.4	yes	yes	65	4.6
160	1	4	4	8.7	yes	yes	63	-	49.0	yes	yes	68	-	8.7	yes	yes	63	6.3
160	1	4	5	31.4	yes	yes	65	-	2.8	yes	yes	63	-	23.2	yes	yes	64	9.4
160	1	8	1	80.5	yes	yes	65	-	20.5	yes	yes	65	-	63.4	yes	yes	65	9.2
160	1	8	2	6.9	yes	yes	62	-	2.6	yes	yes	64	-	7.8	yes	yes	62	8.1
160	1	8	3	2.4	yes	yes	65	-	0.8	yes	yes	63	-	5.6	yes	yes	65	4.6
160	1	8	4	1.4	yes	yes	61	-	18.2	yes	yes	62	-	4.7	yes	yes	61	8.2
160	1	8	5	1.9	yes	yes	64	-	1.3	yes	yes	63	-	5.6	yes	yes	64	4.7
160	2	1	1	-	no	-	-	-	6.0	yes	yes	45	-	63.6	yes	yes	45	4.4
160	2	1	2	-	no	-	-	-	2.3	yes	yes	48	-	18.1	yes	yes	46	2.2
160	2	1	3	-	no	-	-	-	5.7	yes	yes	43	-	25.5	yes	yes	42	0.0
160	2	1	4	-	no	-	-	-	4.1	yes	yes	44	-	17.3	yes	yes	44	2.3
160	2	1	5	-	no	-	-	-	2.5	yes	yes	44	-	18.0	yes	yes	44	2.3
160	2	2	1	373.7	yes	yes	35	-	56.0	yes	yes	34	-	101.0	yes	yes	35	8.6
160	2	2	2	34.3	yes	yes	35	-	40.1	yes	yes	34	-	31.1	yes	yes	35	8.6
160	2	2	3	20.7	yes	yes	35	-	17.7	yes	yes	33	-	47.4	yes	yes	35	8.6
160	2	2	4	6.8	yes	yes	33	-	8.1	yes	yes	35	-	25.5	yes	yes	33	6.1
160	2	2	5	2.9	yes	yes	34	-	7.8	yes	yes	36	-	19.8	yes	yes	34	2.9
160	2	4	1	1800.0	yes	no	32	-	13.2	yes	yes	35	-	180.0	yes	yes	31	16.1
160	2	4	2	11.0	yes	yes	33	-	15.1	yes	yes	34	-	14.2	yes	yes	33	12.1
160	2	4	3	11.5	yes	yes	31	-	14.3	yes	yes	35	-	10.9	yes	yes	31	6.5
160	2	4	4	3.2	yes	yes	33	-	1.1	yes	yes	31	-	11.7	yes	yes	33	9.1
160	2	4	5	2.1	yes	yes	32	-	2.3	yes	yes	32	-	5.9	yes	yes	32	9.4
160	2	8	1	54.2	yes	yes	33	-	349.4	yes	yes	35	-	87.4	yes	yes	33	12.1
160	2	8	2	0.8	yes	yes	31	-	1.6	yes	yes	32	-	2.0	yes	yes	31	6.5
160	2	8	3	1.1	yes	yes	31	-	3.3	yes	yes	31	-	2.5	yes	yes	31	6.5
160	2	8	4	3.4	yes	yes	33	-	3.1	yes	yes	31	-	7.1	yes	yes	32	9.4
160	2	8	5	0.6	yes	yes	32	-	0.6	yes	yes	32	-	3.7	yes	yes	32	9.4
160	3	1	1	-	no	-	-	-	3.5	yes	yes	30	-	90.4	yes	yes	30	10.0
160	3	1	2	-	no	-	-	-	2.7	yes	yes	31	-	38.9	yes	yes	31	6.5
160	3	1	3	-	no	-	-	-	3.4	yes	yes	28	-	40.8	yes	yes	29	3.4
160	3	1	4	-	no	-	-	-	7.5	yes	yes	29	-	58.2	yes	yes	29	3.4
160	3	1	5	-	no	-	-	-	2.9	yes	yes	30	-	20.6	yes	yes	30	6.7
160	3	2	1	200.1	yes	yes	24	-	42.5	yes	yes	24	-	34.3	yes	yes	24	12.5
160	3	2	2	9.3	yes	yes	25	-	15.7	yes	yes	24	-	44.0	yes	yes	25	8.0
160	3	2	3	41.3	yes	yes	22	-	7.5	yes	yes	23	-	29.7	yes	yes	22	9.1
160	3	2	4	4.5	yes	yes	23	-	16.6	yes	yes	22	-	14.4	yes	yes	23	8.7
160	3	2	5	2.4	yes	yes	22	-	3.6	yes	yes	22	-	10.0	yes	yes	22	9.1
160	3	4	1	22.5	yes	yes	24	-	7.8	yes	yes	22	-	1687.5	yes	yes	24	12.5
160	3	4	2	59.5	yes	yes	22	-	94.3	yes	yes	22	-	72.3	yes	yes	21	14.3
160	3	4	3	13.1	yes	yes	21	-	6.1	yes	yes	22	-	24.7	yes	yes	21	9.5
160	3	4	4	4.2	yes	yes	23	-	10.1	yes	yes	23	-	13.1	yes	yes	23	8.7
160	3	4	5	2.5	yes	yes	22	-	5.8	yes	yes	21	-	13.5	yes	yes	22	9.1
160	3	8	1	2.9	yes	yes	21	-	14.5	yes	yes	21	-	6.4	yes	yes	21	4.8
160	3	8	2	446.1	yes	yes	22	-	6.8	yes	yes	22	-	232.9	yes	yes	22	13.6
160	3	8	3	12.5	yes	yes	21	-	5.3	yes	yes	22	-	11.3	yes	yes	21	9.5
160	3	8	4	1.2	yes	yes	23	-	0.9	yes	yes	22	-	22.1	yes	yes	23	4.3
160	3	8	5	1.4	yes	yes	22	-	0.8	yes	yes	22	-	9.4	yes	yes	22	9.1

Table A.16: Problem $V, Q|sd, r_i, d_i|CLT$ (part 4)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
20	1	1	1	-	no	-	-	-	0.0	yes	yes	15	-	0.0	yes	yes	15	13.3
20	1	1	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	15	13.3
20	1	1	3	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	13	15.4
20	1	1	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	15	6.7
20	1	1	5	-	no	-	-	-	0.0	yes	yes	13	-	0.0	yes	yes	12	8.3
20	1	2	1	0.0	yes	yes	13	-	0.0	yes	yes	13	-	0.0	yes	yes	13	15.4
20	1	2	2	-	no	-	-	-	0.0	yes	yes	13	-	0.0	yes	yes	13	15.4
20	1	2	3	-	no	-	-	-	0.0	yes	yes	12	-	0.0	yes	yes	13	23.1
20	1	2	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	11	27.3
20	1	2	5	0.0	yes	yes	11	9.1	0.0	yes	yes	12	-	0.0	yes	yes	11	9.1
20	1	4	1	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	12	16.7
20	1	4	2	-	no	-	-	-	0.0	yes	yes	15	-	0.0	yes	yes	14	7.1
20	1	4	3	0.0	yes	yes	12	-	-	no	-	-	-	0.0	yes	yes	12	16.7
20	1	4	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	11	9.1
20	1	4	5	0.0	yes	yes	10	-	-	no	-	-	-	0.0	yes	yes	10	10.0
20	1	8	1	-	no	-	-	-	0.0	yes	yes	12	-	0.0	yes	yes	11	9.1
20	1	8	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	12	16.7
20	1	8	3	0.0	yes	yes	12	-	0.0	yes	yes	10	10.0	0.0	yes	yes	11	18.2
20	1	8	4	-	no	-	-	-	0.0	yes	yes	13	-	0.0	yes	yes	12	16.7
20	1	8	5	0.0	yes	yes	11	-	0.0	yes	yes	12	-	0.0	yes	yes	11	9.1
20	2	1	1	-	no	-	-	-	0.0	yes	yes	7	-	0.0	yes	yes	8	12.5
20	2	1	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	10	0.0
20	2	1	3	-	no	-	-	-	0.0	yes	yes	7	-	0.0	yes	yes	8	12.5
20	2	1	4	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	9	0.0
20	2	1	5	-	no	-	-	-	0.0	yes	yes	7	-	0.0	yes	yes	7	14.3
20	2	2	1	0.0	yes	yes	7	-	0.0	yes	yes	7	-	0.0	yes	yes	7	14.3
20	2	2	2	0.0	yes	yes	8	-	-	no	-	-	-	0.0	yes	yes	7	14.3
20	2	2	3	0.0	yes	yes	6	-	0.0	yes	yes	6	33.3	0.0	yes	yes	6	16.7
20	2	2	4	0.0	yes	yes	7	-	0.0	yes	yes	7	-	0.0	yes	yes	7	14.3
20	2	2	5	-	no	-	-	-	0.0	yes	yes	6	-	0.0	yes	yes	5	20.0
20	2	4	1	-	no	-	-	-	0.0	yes	yes	6	-	0.0	yes	yes	8	12.5
20	2	4	2	0.0	yes	yes	7	-	0.0	yes	yes	8	-	0.0	yes	yes	7	14.3
20	2	4	3	0.0	yes	yes	7	-	0.0	yes	yes	6	16.7	0.0	yes	yes	6	16.7
20	2	4	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	6	33.3
20	2	4	5	-	no	-	-	-	0.0	yes	yes	7	-	0.0	yes	yes	6	0.0
20	2	8	1	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	0.0
20	2	8	2	0.0	yes	yes	7	-	0.0	yes	yes	7	-	0.0	yes	yes	7	14.3
20	2	8	3	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
20	2	8	4	0.0	yes	yes	7	-	0.0	yes	yes	6	-	0.0	yes	yes	7	14.3
20	2	8	5	0.0	yes	yes	6	-	0.0	yes	yes	6	-	0.0	yes	yes	6	16.7
20	3	1	1	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	6	16.7
20	3	1	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	6	16.7
20	3	1	3	0.0	yes	yes	5	-	-	no	-	-	-	0.0	yes	yes	5	0.0
20	3	1	4	-	no	-	-	-	0.0	yes	yes	6	-	0.1	yes	yes	6	16.7
20	3	1	5	-	no	-	-	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
20	3	2	1	-	no	-	-	-	0.0	yes	yes	5	-	0.0	yes	yes	5	0.0
20	3	2	2	-	no	-	-	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
20	3	2	3	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	3	2	4	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0
20	3	2	5	0.0	yes	yes	4	-	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	3	4	1	0.0	yes	yes	5	-	0.0	yes	yes	5	0.0	0.0	yes	yes	5	20.0
20	3	4	2	-	no	-	-	-	0.0	yes	yes	5	-	0.0	yes	yes	6	0.0
20	3	4	3	0.0	yes	yes	5	-	0.0	yes	yes	5	20.0	0.0	yes	yes	4	50.0
20	3	4	4	0.0	yes	yes	5	-	0.0	yes	yes	4	-	0.0	yes	yes	5	40.0
20	3	4	5	0.0	yes	yes	5	-	0.0	yes	yes	4	25.0	0.0	yes	yes	5	0.0
20	3	8	1	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	0.0
20	3	8	2	0.0	yes	yes	4	-	0.0	yes	yes	5	-	0.0	yes	yes	4	25.0
20	3	8	3	0.0	yes	yes	4	25.0	0.0	yes	yes	4	-	0.0	yes	yes	4	25.0
20	3	8	4	0.0	yes	yes	5	-	0.0	yes	yes	5	-	0.0	yes	yes	5	40.0
20	3	8	5	0.0	yes	yes	4	-	0.0	yes	yes	5	-	0.0	yes	yes	4	25.0

Table A.17: Problem $V, Q|sd, r_i, d_i|CLT$ (part 5)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
40	1	1	1	-	no	-	-	-	0.1	yes	yes	27	-	0.1	yes	yes	24	8.3
40	1	1	2	0.0	yes	yes	26	-	0.0	yes	yes	25	-	0.1	yes	yes	25	12.0
40	1	1	3	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	31	3.2
40	1	1	4	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	27	3.7
40	1	1	5	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	28	3.6
40	1	2	1	0.0	yes	yes	22	-	0.0	yes	yes	21	-	0.1	yes	yes	21	14.3
40	1	2	2	0.0	yes	yes	22	-	0.0	yes	yes	22	-	0.0	yes	yes	21	4.8
40	1	2	3	0.1	yes	yes	23	-	0.1	yes	yes	23	-	0.1	yes	yes	22	9.1
40	1	2	4	0.0	yes	yes	24	-	0.0	yes	yes	23	-	0.1	yes	yes	22	4.5
40	1	2	5	-	no	-	-	-	0.0	yes	yes	24	-	0.1	yes	yes	22	4.5
40	1	4	1	-	no	-	-	-	0.0	yes	yes	21	-	0.1	yes	yes	21	9.5
40	1	4	2	0.0	yes	yes	24	-	0.0	yes	yes	22	-	0.1	yes	yes	22	9.1
40	1	4	3	0.0	yes	yes	23	-	0.0	yes	yes	24	-	0.1	yes	yes	22	9.1
40	1	4	4	0.0	yes	yes	22	-	0.1	yes	yes	24	-	0.1	yes	yes	21	14.3
40	1	4	5	0.0	yes	yes	25	-	0.0	yes	yes	26	-	0.1	yes	yes	24	8.3
40	1	8	1	0.0	yes	yes	21	-	0.0	yes	yes	21	-	0.0	yes	yes	21	0.0
40	1	8	2	0.0	yes	yes	20	-	0.0	yes	yes	21	-	0.1	yes	yes	19	21.1
40	1	8	3	0.0	yes	yes	23	-	0.1	yes	yes	23	-	0.1	yes	yes	22	18.2
40	1	8	4	0.0	yes	yes	21	-	0.0	yes	yes	22	-	0.0	yes	yes	21	14.3
40	1	8	5	0.0	yes	yes	21	-	0.0	yes	yes	22	-	0.0	yes	yes	21	0.0
40	2	1	1	0.0	yes	yes	13	-	0.0	yes	yes	13	-	0.1	yes	yes	12	16.7
40	2	1	2	-	no	-	-	-	-	no	-	-	-	0.2	yes	yes	14	0.0
40	2	1	3	-	no	-	-	-	-	no	-	-	-	0.2	yes	yes	15	13.3
40	2	1	4	-	no	-	-	-	0.0	yes	yes	14	-	0.1	yes	yes	15	6.7
40	2	1	5	-	no	-	-	-	0.0	yes	yes	14	-	0.2	yes	yes	14	7.1
40	2	2	1	0.0	yes	yes	12	-	0.0	yes	yes	12	-	0.1	yes	yes	12	8.3
40	2	2	2	-	no	-	-	-	0.0	yes	yes	12	-	0.1	yes	yes	11	18.2
40	2	2	3	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.1	yes	yes	11	9.1
40	2	2	4	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.0	yes	yes	11	9.1
40	2	2	5	0.0	yes	yes	12	-	0.0	yes	yes	11	-	0.1	yes	yes	11	9.1
40	2	4	1	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.1	yes	yes	11	18.2
40	2	4	2	-	no	-	-	-	0.0	yes	yes	11	-	0.1	yes	yes	11	9.1
40	2	4	3	0.0	yes	yes	12	-	0.0	yes	yes	11	-	0.1	yes	yes	12	8.3
40	2	4	4	0.0	yes	yes	11	-	0.0	yes	yes	13	-	0.1	yes	yes	11	18.2
40	2	4	5	0.0	yes	yes	13	-	0.0	yes	yes	11	-	0.1	yes	yes	13	7.7
40	2	8	1	0.0	yes	yes	11	-	0.0	yes	yes	11	-	0.1	yes	yes	11	18.2
40	2	8	2	0.0	yes	yes	11	-	0.0	yes	yes	12	-	0.1	yes	yes	11	9.1
40	2	8	3	-	no	-	-	-	0.0	yes	yes	11	-	0.1	yes	yes	11	9.1
40	2	8	4	0.0	yes	yes	12	-	0.0	yes	yes	11	-	0.1	yes	yes	12	16.7
40	2	8	5	0.0	yes	yes	10	10.0	0.0	yes	yes	11	-	0.1	yes	yes	10	20.0
40	3	1	1	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.1	yes	yes	8	25.0
40	3	1	2	-	no	-	-	-	0.0	yes	yes	10	-	0.1	yes	yes	9	11.1
40	3	1	3	-	no	-	-	-	0.0	yes	yes	11	-	0.3	yes	yes	10	10.0
40	3	1	4	-	no	-	-	-	0.1	yes	yes	8	-	0.2	yes	yes	9	11.1
40	3	1	5	-	no	-	-	-	0.0	yes	yes	10	-	0.2	yes	yes	9	11.1
40	3	2	1	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.1	yes	yes	8	12.5
40	3	2	2	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.1	yes	yes	8	12.5
40	3	2	3	0.0	yes	yes	9	-	0.0	yes	yes	9	-	0.2	yes	yes	8	25.0
40	3	2	4	0.0	yes	yes	8	-	0.0	yes	yes	9	-	0.1	yes	yes	8	12.5
40	3	2	5	0.0	yes	yes	9	-	0.0	yes	yes	8	-	0.1	yes	yes	9	11.1
40	3	4	1	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.1	yes	yes	8	12.5
40	3	4	2	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.1	yes	yes	8	12.5
40	3	4	3	0.1	yes	yes	9	-	0.0	yes	yes	8	-	0.1	yes	yes	8	25.0
40	3	4	4	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.1	yes	yes	8	0.0
40	3	4	5	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.1	yes	yes	8	12.5
40	3	8	1	0.0	yes	yes	7	-	0.0	yes	yes	8	-	0.1	yes	yes	7	14.3
40	3	8	2	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.1	yes	yes	8	12.5
40	3	8	3	0.0	yes	yes	8	-	0.0	yes	yes	7	-	0.1	yes	yes	8	12.5
40	3	8	4	0.0	yes	yes	8	-	0.0	yes	yes	8	-	0.1	yes	yes	8	25.0
40	3	8	5	0.0	yes	yes	8	-	0.0	yes	yes	9	-	0.1	yes	yes	8	25.0

Table A.18: Problem $V, Q|sd, r_i, d_i|CLT$ (part 6)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
80	1	1	1	-	no	-	-	-	-	no	-	-	-	1.2	yes	yes	46	8.7
80	1	1	2	0.1	yes	yes	47	-	0.3	yes	yes	46	-	0.8	yes	yes	46	10.9
80	1	1	3	-	no	-	-	-	0.5	yes	yes	49	-	0.8	yes	yes	50	10.0
80	1	1	4	-	no	-	-	-	0.2	yes	yes	52	-	0.6	yes	yes	50	6.0
80	1	1	5	0.1	yes	yes	48	-	0.1	yes	yes	48	-	0.6	yes	yes	48	2.1
80	1	2	1	0.1	yes	yes	42	-	0.1	yes	yes	42	-	0.4	yes	yes	42	11.9
80	1	2	2	0.1	yes	yes	45	-	0.4	yes	yes	42	-	0.3	yes	yes	45	2.2
80	1	2	3	0.2	yes	yes	45	-	0.5	yes	yes	45	-	0.6	yes	yes	45	8.9
80	1	2	4	0.1	yes	yes	44	-	0.4	yes	yes	42	-	0.3	yes	yes	44	4.5
80	1	2	5	0.1	yes	yes	42	-	0.1	yes	yes	42	-	0.2	yes	yes	41	4.9
80	1	4	1	0.1	yes	yes	42	-	0.3	yes	yes	43	-	0.5	yes	yes	42	2.4
80	1	4	2	0.1	yes	yes	41	-	0.2	yes	yes	40	-	0.1	yes	yes	41	2.4
80	1	4	3	0.1	yes	yes	41	-	0.1	yes	yes	43	-	0.2	yes	yes	41	0.0
80	1	4	4	0.3	yes	yes	40	-	0.4	yes	yes	42	-	0.3	yes	yes	40	17.5
80	1	4	5	0.1	yes	yes	42	-	0.1	yes	yes	41	-	0.3	yes	yes	42	11.9
80	1	8	1	0.1	yes	yes	42	-	0.1	yes	yes	41	-	0.2	yes	yes	42	7.1
80	1	8	2	0.1	yes	yes	42	-	0.0	yes	yes	42	-	0.2	yes	yes	42	2.4
80	1	8	3	0.1	yes	yes	44	-	0.1	yes	yes	41	-	0.5	yes	yes	44	9.1
80	1	8	4	0.1	yes	yes	41	-	0.2	yes	yes	42	-	0.2	yes	yes	41	4.9
80	1	8	5	0.0	yes	yes	40	-	0.1	yes	yes	40	-	0.1	yes	yes	40	0.0
80	2	1	1	0.2	yes	yes	23	-	0.3	yes	yes	25	-	1.0	yes	yes	23	4.3
80	2	1	2	0.2	yes	yes	24	-	0.2	yes	yes	23	-	1.7	yes	yes	23	8.7
80	2	1	3	-	no	-	-	-	0.3	yes	yes	26	-	2.1	yes	yes	24	8.3
80	2	1	4	0.2	yes	yes	25	-	0.7	yes	yes	26	-	1.0	yes	yes	25	8.0
80	2	1	5	0.2	yes	yes	24	-	0.2	yes	yes	24	-	0.9	yes	yes	24	8.3
80	2	2	1	0.1	yes	yes	22	-	0.2	yes	yes	22	4.5	0.6	yes	yes	22	0.0
80	2	2	2	0.1	yes	yes	22	-	0.1	yes	yes	22	-	0.5	yes	yes	22	4.5
80	2	2	3	0.2	yes	yes	21	-	0.2	yes	yes	22	-	0.8	yes	yes	21	14.3
80	2	2	4	0.3	yes	yes	21	-	0.2	yes	yes	22	-	0.6	yes	yes	21	9.5
80	2	2	5	0.1	yes	yes	25	-	0.1	yes	yes	22	-	0.9	yes	yes	25	0.0
80	2	4	1	0.1	yes	yes	22	-	0.1	yes	yes	21	-	0.3	yes	yes	22	4.5
80	2	4	2	0.1	yes	yes	21	-	0.1	yes	yes	21	-	0.2	yes	yes	21	4.8
80	2	4	3	0.1	yes	yes	22	4.5	0.1	yes	yes	21	-	0.4	yes	yes	21	9.5
80	2	4	4	0.2	yes	yes	21	-	0.1	yes	yes	23	-	0.5	yes	yes	21	4.8
80	2	4	5	0.1	yes	yes	21	-	0.1	yes	yes	21	-	0.4	yes	yes	21	4.8
80	2	8	1	0.1	yes	yes	21	-	0.1	yes	yes	21	-	0.3	yes	yes	21	9.5
80	2	8	2	0.1	yes	yes	22	-	0.1	yes	yes	21	-	0.4	yes	yes	22	4.5
80	2	8	3	0.1	yes	yes	21	-	0.2	yes	yes	22	-	0.4	yes	yes	21	4.8
80	2	8	4	0.1	yes	yes	21	-	0.1	yes	yes	21	-	0.2	yes	yes	21	4.8
80	2	8	5	0.1	yes	yes	22	-	0.1	yes	yes	22	-	0.4	yes	yes	22	4.5
80	3	1	1	-	no	-	-	-	0.4	yes	yes	17	-	1.3	yes	yes	16	18.8
80	3	1	2	0.1	yes	yes	17	-	0.2	yes	yes	16	-	1.2	yes	yes	17	5.9
80	3	1	3	-	no	-	-	-	0.3	yes	yes	17	-	1.9	yes	yes	18	11.1
80	3	1	4	-	no	-	-	-	0.2	yes	yes	19	-	1.4	yes	yes	17	0.0
80	3	1	5	0.1	yes	yes	16	-	0.1	yes	yes	16	-	1.0	yes	yes	16	6.3
80	3	2	1	0.1	yes	yes	15	-	0.2	yes	yes	15	-	0.7	yes	yes	15	13.3
80	3	2	2	0.1	yes	yes	15	-	0.1	yes	yes	15	-	0.6	yes	yes	15	20.0
80	3	2	3	0.1	yes	yes	15	-	0.1	yes	yes	14	-	0.3	yes	yes	15	0.0
80	3	2	4	0.1	yes	yes	14	-	0.1	yes	yes	15	-	0.3	yes	yes	14	7.1
80	3	2	5	0.1	yes	yes	16	-	0.1	yes	yes	15	-	0.4	yes	yes	16	6.3
80	3	4	1	0.1	yes	yes	15	-	0.1	yes	yes	15	-	0.4	yes	yes	15	13.3
80	3	4	2	0.0	yes	yes	15	-	0.0	yes	yes	15	-	0.3	yes	yes	15	0.0
80	3	4	3	0.1	yes	yes	15	-	0.1	yes	yes	15	-	0.3	yes	yes	15	0.0
80	3	4	4	0.1	yes	yes	15	-	0.1	yes	yes	15	-	0.3	yes	yes	15	6.7
80	3	4	5	0.1	yes	yes	15	-	0.2	yes	yes	16	-	0.5	yes	yes	15	6.7
80	3	8	1	0.1	yes	yes	15	-	0.1	yes	yes	15	-	0.3	yes	yes	15	0.0
80	3	8	2	0.0	yes	yes	15	-	0.1	yes	yes	15	-	0.3	yes	yes	15	0.0
80	3	8	3	0.1	yes	yes	15	-	0.1	yes	yes	15	-	0.5	yes	yes	15	6.7
80	3	8	4	0.1	yes	yes	14	-	0.1	yes	yes	14	-	0.3	yes	yes	14	7.1
80	3	8	5	0.1	yes	yes	16	-	0.1	yes	yes	15	-	1.1	yes	yes	16	6.3

Table A.19: Problem $V, Q|sd, r_i, d_i|CLT$ (part 7)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)	CPU(s)	feas	opt	CLT	gapGr(%)
160	1	1	1	6.3	yes	yes	92	-	5.0	yes	yes	95	-	35.4	yes	yes	91	5.5
160	1	1	2	1.5	yes	yes	99	-	5.8	yes	yes	92	-	8.6	yes	yes	99	3.0
160	1	1	3	1.5	yes	yes	84	-	4.8	yes	yes	87	-	11.3	yes	yes	83	8.4
160	1	1	4	2.1	yes	yes	92	-	2.5	yes	yes	92	-	20.8	yes	yes	93	9.7
160	1	1	5	2.1	yes	yes	90	-	4.3	yes	yes	90	-	8.9	yes	yes	90	5.6
160	1	2	1	4.2	yes	yes	88	-	12.4	yes	yes	82	-	13.0	yes	yes	88	3.4
160	1	2	2	0.7	yes	yes	85	-	2.1	yes	yes	86	-	2.4	yes	yes	85	3.5
160	1	2	3	2.6	yes	yes	82	-	0.7	yes	yes	83	-	7.0	yes	yes	82	6.1
160	1	2	4	1.1	yes	yes	82	-	0.8	yes	yes	82	-	2.3	yes	yes	82	2.4
160	1	2	5	0.3	yes	yes	81	-	1.8	yes	yes	85	-	0.6	yes	yes	81	1.2
160	1	4	1	0.3	yes	yes	82	-	0.4	yes	yes	82	-	1.3	yes	yes	82	2.4
160	1	4	2	0.5	yes	yes	83	-	0.4	yes	yes	81	-	1.1	yes	yes	83	1.2
160	1	4	3	0.3	yes	yes	83	-	1.0	yes	yes	80	-	2.0	yes	yes	83	1.2
160	1	4	4	0.9	yes	yes	79	-	0.4	yes	yes	82	-	2.8	yes	yes	79	5.1
160	1	4	5	0.3	yes	yes	81	-	0.2	yes	yes	82	-	1.2	yes	yes	81	2.5
160	1	8	1	0.3	yes	yes	83	-	0.7	yes	yes	82	-	2.4	yes	yes	82	3.7
160	1	8	2	0.3	yes	yes	81	-	0.2	yes	yes	81	-	1.2	yes	yes	81	1.2
160	1	8	3	0.2	yes	yes	81	-	0.3	yes	yes	83	-	1.0	yes	yes	81	0.0
160	1	8	4	0.3	yes	yes	84	-	0.4	yes	yes	81	-	1.3	yes	yes	84	1.2
160	1	8	5	0.3	yes	yes	81	-	0.3	yes	yes	80	-	1.0	yes	yes	81	2.5
160	2	1	1	4.2	yes	yes	47	-	5.8	yes	yes	47	-	30.4	yes	yes	47	6.4
160	2	1	2	0.8	yes	yes	49	-	3.1	yes	yes	48	-	8.2	yes	yes	49	4.1
160	2	1	3	0.8	yes	yes	45	-	4.6	yes	yes	43	-	5.4	yes	yes	45	6.7
160	2	1	4	1.4	yes	yes	47	-	2.4	yes	yes	45	-	11.5	yes	yes	47	4.3
160	2	1	5	1.4	yes	yes	44	-	1.7	yes	yes	46	-	5.8	yes	yes	44	4.5
160	2	2	1	1.5	yes	yes	44	-	6.8	yes	yes	44	-	2.9	yes	yes	44	4.5
160	2	2	2	0.8	yes	yes	42	-	1.3	yes	yes	42	-	3.8	yes	yes	42	7.1
160	2	2	3	0.6	yes	yes	42	-	1.5	yes	yes	40	-	1.5	yes	yes	42	2.4
160	2	2	4	0.6	yes	yes	42	-	0.7	yes	yes	41	-	2.8	yes	yes	42	4.8
160	2	2	5	0.5	yes	yes	42	-	0.4	yes	yes	42	-	2.0	yes	yes	41	2.4
160	2	4	1	0.7	yes	yes	41	-	1.7	yes	yes	43	-	2.5	yes	yes	41	4.9
160	2	4	2	0.9	yes	yes	43	-	0.9	yes	yes	41	-	2.6	yes	yes	43	2.3
160	2	4	3	0.6	yes	yes	43	-	1.0	yes	yes	42	-	2.8	yes	yes	43	4.7
160	2	4	4	0.2	yes	yes	42	-	0.5	yes	yes	42	-	2.2	yes	yes	42	2.4
160	2	4	5	0.3	yes	yes	41	-	0.4	yes	yes	41	-	2.1	yes	yes	41	2.4
160	2	8	1	0.9	yes	yes	41	-	1.8	yes	yes	41	-	9.0	yes	yes	41	7.3
160	2	8	2	0.3	yes	yes	42	-	0.4	yes	yes	42	-	2.7	yes	yes	42	4.8
160	2	8	3	0.2	yes	yes	41	-	0.3	yes	yes	41	2.4	1.0	yes	yes	41	0.0
160	2	8	4	0.2	yes	yes	40	-	0.3	yes	yes	41	-	2.2	yes	yes	40	5.0
160	2	8	5	0.2	yes	yes	40	-	0.3	yes	yes	41	-	1.0	yes	yes	40	2.5
160	3	1	1	1.9	yes	yes	31	-	6.1	yes	yes	31	-	28.4	yes	yes	30	6.7
160	3	1	2	2.4	yes	yes	31	-	3.2	yes	yes	33	-	21.9	yes	yes	31	9.7
160	3	1	3	1.4	yes	yes	30	-	1.3	yes	yes	29	-	17.7	yes	yes	30	13.3
160	3	1	4	2.0	yes	yes	29	-	2.3	yes	yes	31	-	20.0	yes	yes	29	10.3
160	3	1	5	0.7	yes	yes	30	-	1.0	yes	yes	30	-	17.5	yes	yes	30	6.7
160	3	2	1	0.9	yes	yes	28	-	2.0	yes	yes	29	-	8.8	yes	yes	28	7.1
160	3	2	2	0.7	yes	yes	28	-	2.3	yes	yes	29	-	4.1	yes	yes	28	7.1
160	3	2	3	0.9	yes	yes	27	-	0.5	yes	yes	28	-	5.8	yes	yes	27	7.4
160	3	2	4	0.5	yes	yes	27	-	0.7	yes	yes	28	-	3.2	yes	yes	27	3.7
160	3	2	5	0.2	yes	yes	28	-	0.4	yes	yes	29	-	1.4	yes	yes	28	3.6
160	3	4	1	0.3	yes	yes	28	-	0.6	yes	yes	28	-	1.6	yes	yes	28	0.0
160	3	4	2	3.6	yes	yes	29	-	0.6	yes	yes	29	-	5.9	yes	yes	29	0.0
160	3	4	3	0.5	yes	yes	28	-	0.1	yes	yes	28	-	3.0	yes	yes	28	3.6
160	3	4	4	0.2	yes	yes	28	-	0.5	yes	yes	28	-	1.6	yes	yes	28	0.0
160	3	4	5	0.3	yes	yes	28	-	0.4	yes	yes	29	-	4.5	yes	yes	28	3.6
160	3	8	1	0.4	yes	yes	28	-	0.4	yes	yes	27	3.7	4.3	yes	yes	28	3.6
160	3	8	2	0.4	yes	yes	28	-	0.5	yes	yes	29	-	4.7	yes	yes	28	3.6
160	3	8	3	0.3	yes	yes	28	-	0.3	yes	yes	27	-	3.1	yes	yes	28	0.0
160	3	8	4	0.3	yes	yes	27	-	0.4	yes	yes	28	-	3.4	yes	yes	27	3.7
160	3	8	5	0.1	yes	yes	27	-	0.3	yes	yes	28	-	1.5	yes	yes	27	3.7

Table A.20: Problem $V, Q|sd, r_i, d_i|CLT$ (part 8)

Appendix B

Detailed computational results of Chapter 4

In section 4.6 we reported the results of the resolution of formulation (4.2)-(4.5). Tables 4.1-4.6 were calculated by taking the average over the 5 instances with the same value of n, V and Q. In this appendix we report the detailed results of those experiments. In all tables, column opt reports if the calculated solution value is optimal for the instance. All other column headings have the same meaning of tables 4.2-4.5.

n	V	Q	instance	CPU(s)	$\sum C_i$	opt	gapGr(%)
20	1	1	1	0.0	117.0	yes	6.8
20	1	1	2	0.0	140.0	yes	0.0
20	1	1	3	0.0	106.0	yes	2.8
20	1	1	4	0.0	110.0	yes	13.6
20	1	1	5	0.0	99.0	yes	8.1
20	1	2	1	0.0	70.0	yes	5.7
20	1	2	2	0.0	82.0	yes	0.0
20	1	2	3	0.0	60.0	yes	5.0
20	1	2	4	0.0	65.0	yes	9.2
20	1	2	5	0.0	56.0	yes	7.1
20	1	4	1	0.0	46.0	yes	2.2
20	1	4	2	0.0	54.0	yes	0.0
20	1	4	3	0.0	39.0	yes	10.3
20	1	4	4	0.0	43.0	yes	0.0
20	1	4	5	0.0	35.0	yes	8.6
20	1	8	1	0.0	36.0	yes	0.0
20	1	8	2	0.0	40.0	yes	0.0
20	1	8	3	0.0	30.0	yes	3.3
20	1	8	4	0.0	34.0	yes	0.0
20	1	8	5	0.0	26.0	yes	3.8
20	2	1	1	0.0	70.0	yes	5.7
20	2	1	2	0.0	82.0	yes	0.0
20	2	1	3	0.0	60.0	yes	5.0
20	2	1	4	0.0	65.0	yes	9.2
20	2	1	5	0.0	56.0	yes	7.1
20	2	2	1	0.0	46.0	yes	2.2
20	2	2	2	0.0	54.0	yes	0.0
20	2	2	3	0.0	39.0	yes	10.3
20	2	2	4	0.0	43.0	yes	0.0
20	2	2	5	0.0	35.0	yes	8.6
20	2	4	1	0.0	36.0	yes	0.0
20	2	4	2	0.0	40.0	yes	0.0
20	2	4	3	0.0	30.0	yes	3.3
20	2	4	4	0.0	34.0	yes	0.0
20	2	4	5	0.0	26.0	yes	3.8
20	2	8	1	0.0	31.0	yes	0.0
20	2	8	2	0.0	33.0	yes	0.0
20	2	8	3	0.0	26.0	yes	0.0
20	2	8	4	0.0	30.0	yes	0.0
20	2	8	5	0.0	23.0	yes	0.0
20	3	1	1	0.0	54.0	yes	3.7
20	3	1	2	0.0	63.0	yes	0.0
20	3	1	3	0.0	46.0	yes	6.5
20	3	1	4	0.0	50.0	yes	6.0
20	3	1	5	0.0	42.0	yes	7.1
20	3	2	1	0.0	39.0	yes	0.0
20	3	2	2	0.0	45.0	yes	0.0
20	3	2	3	0.0	32.0	yes	3.1
20	3	2	4	0.0	36.0	yes	0.0
20	3	2	5	0.0	28.0	yes	3.6
20	3	4	1	0.0	32.0	yes	0.0
20	3	4	2	0.0	36.0	yes	0.0
20	3	4	3	0.0	26.0	yes	0.0
20	3	4	4	0.0	30.0	yes	0.0
20	3	4	5	0.0	23.0	yes	0.0
20	3	8	1	0.0	31.0	yes	0.0
20	3	8	2	0.0	33.0	yes	0.0
20	3	8	3	0.0	26.0	yes	0.0
20	3	8	4	0.0	30.0	yes	0.0
20	3	8	5	0.0	23.0	yes	0.0

n	V	Q	instance	CPU(s)	$\sum C_i$	opt	gapGr(%)
40	1	1	1	0.1	346.0	yes	4.9
40	1	1	2	0.1	379.0	yes	4.0
40	1	1	3	0.1	476.0	yes	6.7
40	1	1	4	0.1	419.0	yes	3.3
40	1	1	5	0.1	400.0	yes	9.3
40	1	2	1	0.1	191.0	yes	7.9
40	1	2	2	0.1	201.0	yes	3.0
40	1	2	3	0.0	255.0	yes	5.9
40	1	2	4	0.0	222.0	yes	3.6
40	1	2	5	0.0	212.0	yes	6.6
40	1	4	1	0.1	116.0	yes	4.3
40	1	4	2	0.0	118.0	yes	0.8
40	1	4	3	0.0	145.0	yes	8.3
40	1	4	4	0.0	125.0	yes	3.2
40	1	4	5	0.0	119.0	yes	3.4
40	1	8	1	0.0	80.0	yes	1.3
40	1	8	2	0.0	80.0	yes	1.3
40	1	8	3	0.0	91.0	yes	4.4
40	1	8	4	0.0	78.0	yes	0.0
40	1	8	5	0.0	76.0	yes	0.0
40	2	1	1	0.1	191.0	yes	7.9
40	2	1	2	0.1	201.0	yes	3.0
40	2	1	3	0.0	255.0	yes	5.9
40	2	1	4	0.0	222.0	yes	3.6
40	2	1	5	0.1	212.0	yes	6.6
40	2	2	1	0.1	116.0	yes	4.3
40	2	2	2	0.0	118.0	yes	0.8
40	2	2	3	0.0	145.0	yes	8.3
40	2	2	4	0.0	125.0	yes	3.2
40	2	2	5	0.0	119.0	yes	3.4
40	2	4	1	0.0	80.0	yes	1.3
40	2	4	2	0.0	80.0	yes	1.3
40	2	4	3	0.0	91.0	yes	4.4
40	2	4	4	0.0	78.0	yes	0.0
40	2	4	5	0.0	76.0	yes	0.0
40	2	8	1	0.0	63.0	yes	0.0
40	2	8	2	0.0	65.0	yes	0.0
40	2	8	3	0.0	66.0	yes	0.0
40	2	8	4	0.0	56.0	yes	0.0
40	2	8	5	0.0	58.0	yes	0.0
40	3	1	1	0.1	141.0	yes	7.1
40	3	1	2	0.1	145.0	yes	2.8
40	3	1	3	0.1	181.0	yes	8.3
40	3	1	4	0.1	157.0	yes	3.2
40	3	1	5	0.1	150.0	yes	4.7
40	3	2	1	0.0	91.0	yes	3.3
40	3	2	2	0.0	92.0	yes	0.0
40	3	2	3	0.0	108.0	yes	6.5
40	3	2	4	0.0	93.0	yes	2.2
40	3	2	5	0.0	89.0	yes	3.4
40	3	4	1	0.0	68.0	yes	1.5
40	3	4	2	0.0	69.0	yes	0.0
40	3	4	3	0.0	74.0	yes	0.0
40	3	4	4	0.0	63.0	yes	0.0
40	3	4	5	0.0	63.0	yes	0.0
40	3	8	1	0.0	59.0	yes	0.0
40	3	8	2	0.0	58.0	yes	0.0
40	3	8	3	0.0	58.0	yes	0.0
40	3	8	4	0.0	51.0	yes	0.0
40	3	8	5	0.0	55.0	yes	0.0

Table B.1: Problem $V, Q|sd, u|\sum C_i$ (part 1)

n	V	Q	instance	CPU(s)	$\sum C_i$	opt	gapGr(%)
80	1	1	1	0.4	1345.0	yes	3.4
80	1	1	2	0.4	1304.0	yes	2.1
80	1	1	3	0.3	1453.0	yes	6.5
80	1	1	4	0.4	1559.0	yes	7.4
80	1	1	5	0.3	1360.0	yes	4.7
80	1	2	1	0.3	703.0	yes	2.8
80	1	2	2	0.3	681.0	yes	2.2
80	1	2	3	0.1	767.0	yes	6.5
80	1	2	4	0.3	809.0	yes	7.2
80	1	2	5	0.1	706.0	yes	5.1
80	1	4	1	0.1	383.0	yes	2.9
80	1	4	2	0.1	371.0	yes	3.8
80	1	4	3	0.1	425.0	yes	5.4
80	1	4	4	0.1	435.0	yes	8.0
80	1	4	5	0.1	380.0	yes	5.0
80	1	8	1	0.1	225.0	yes	4.4
80	1	8	2	0.1	223.0	yes	4.9
80	1	8	3	0.1	256.0	yes	3.5
80	1	8	4	0.1	249.0	yes	7.2
80	1	8	5	0.1	221.0	yes	3.2
80	2	1	1	0.4	703.0	yes	2.8
80	2	1	2	0.3	681.0	yes	2.2
80	2	1	3	0.4	767.0	yes	6.5
80	2	1	4	0.3	809.0	yes	7.2
80	2	1	5	0.4	706.0	yes	5.1
80	2	2	1	0.1	383.0	yes	2.9
80	2	2	2	0.3	371.0	yes	3.8
80	2	2	3	0.2	425.0	yes	5.4
80	2	2	4	0.2	435.0	yes	8.0
80	2	2	5	0.2	380.0	yes	5.0
80	2	4	1	0.1	225.0	yes	4.4
80	2	4	2	0.1	223.0	yes	4.9
80	2	4	3	0.1	256.0	yes	3.5
80	2	4	4	0.1	249.0	yes	7.2
80	2	4	5	0.1	221.0	yes	3.2
80	2	8	1	0.1	151.0	yes	6.0
80	2	8	2	0.1	154.0	yes	1.3
80	2	8	3	0.1	173.0	yes	4.6
80	2	8	4	0.1	158.0	yes	5.7
80	2	8	5	0.1	143.0	yes	2.8
80	3	1	1	0.4	489.0	yes	2.7
80	3	1	2	0.4	475.0	yes	2.3
80	3	1	3	0.3	539.0	yes	6.5
80	3	1	4	0.3	559.0	yes	7.5
80	3	1	5	0.3	488.0	yes	5.5
80	3	2	1	0.3	278.0	yes	3.2
80	3	2	2	0.2	272.0	yes	4.8
80	3	2	3	0.2	311.0	yes	6.4
80	3	2	4	0.2	311.0	yes	8.0
80	3	2	5	0.2	273.0	yes	4.4
80	3	4	1	0.1	174.0	yes	5.7
80	3	4	2	0.1	176.0	yes	2.8
80	3	4	3	0.1	201.0	yes	2.5
80	3	4	4	0.1	187.0	yes	6.4
80	3	4	5	0.1	169.0	yes	3.0
80	3	8	1	0.1	126.0	yes	3.2
80	3	8	2	0.1	129.0	yes	3.1
80	3	8	3	0.1	146.0	yes	2.7
80	3	8	4	0.1	129.0	yes	3.1
80	3	8	5	0.1	121.0	yes	0.8

n	V	Q	instance	CPU(s)	$\sum C_i$	opt	gapGr(%)
160	1	1	1	7.3	5271.0	yes	1.2
160	1	1	2	2.4	5246.0	yes	1.3
160	1	1	3	2.1	4736.0	yes	0.6
160	1	1	4	1.9	4912.0	yes	2.8
160	1	1	5	1.4	4906.0	yes	1.3
160	1	2	1	1.5	2688.0	yes	1.4
160	1	2	2	1.4	2676.0	yes	1.2
160	1	2	3	1.9	2433.0	yes	0.6
160	1	2	4	1.1	2517.0	yes	2.5
160	1	2	5	0.9	2511.0	yes	1.1
160	1	4	1	1.5	1399.0	yes	2.4
160	1	4	2	0.7	1394.0	yes	1.3
160	1	4	3	0.5	1284.0	yes	1.0
160	1	4	4	1.0	1325.0	yes	2.6
160	1	4	5	0.6	1314.0	yes	1.6
160	1	8	1	0.3	765.0	yes	3.1
160	1	8	2	0.3	756.0	yes	1.2
160	1	8	3	0.3	713.0	yes	2.0
160	1	8	4	0.2	732.0	yes	3.7
160	1	8	5	0.2	721.0	yes	1.9
160	2	1	1	3.0	2688.0	yes	1.4
160	2	1	2	6.0	2676.0	yes	1.2
160	2	1	3	3.9	2433.0	yes	0.6
160	2	1	4	1.8	2517.0	yes	2.5
160	2	1	5	1.7	2511.0	yes	1.1
160	2	2	1	1.4	1399.0	yes	2.4
160	2	2	2	1.2	1394.0	yes	1.3
160	2	2	3	1.1	1284.0	yes	1.0
160	2	2	4	1.2	1325.0	yes	2.6
160	2	2	5	0.9	1314.0	yes	1.6
160	2	4	1	1.0	765.0	yes	3.1
160	2	4	2	0.9	756.0	yes	1.2
160	2	4	3	0.9	713.0	yes	2.0
160	2	4	4	0.9	732.0	yes	3.7
160	2	4	5	0.8	721.0	yes	1.9
160	2	8	1	0.6	458.0	yes	4.1
160	2	8	2	0.6	442.0	yes	1.8
160	2	8	3	0.4	431.0	yes	1.9
160	2	8	4	0.4	442.0	yes	4.8
160	2	8	5	0.4	433.0	yes	1.2
160	3	1	1	2.6	1828.0	yes	1.6
160	3	1	2	2.2	1820.0	yes	1.3
160	3	1	3	2.9	1666.0	yes	0.8
160	3	1	4	2.5	1721.0	yes	2.3
160	3	1	5	1.8	1713.0	yes	1.1
160	3	2	1	1.7	974.0	yes	2.7
160	3	2	2	1.2	968.0	yes	1.2
160	3	2	3	1.3	903.0	yes	1.4
160	3	2	4	1.1	930.0	yes	2.4
160	3	2	5	0.9	918.0	yes	1.9
160	3	4	1	1.0	561.0	yes	3.7
160	3	4	2	1.2	548.0	yes	1.8
160	3	4	3	1.0	524.0	yes	1.9
160	3	4	4	0.8	538.0	yes	4.1
160	3	4	5	1.2	528.0	yes	1.7
160	3	8	1	0.6	359.0	yes	2.5
160	3	8	2	0.3	344.0	yes	1.5
160	3	8	3	0.2	342.0	yes	0.9
160	3	8	4	0.3	349.0	yes	1.7
160	3	8	5	0.4	338.0	yes	0.3

Table B.2: Problem $V, Q|sd, u|\sum C_i$ (part 2)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
20	1	1	1	0.1	yes	yes	205.0	-	0.0	yes	yes	201.0	-	0.0	yes	yes	205.0	9.8
20	1	1	2	-	no	-	-	-	0.0	yes	yes	176.0	-	0.0	yes	yes	204.0	4.4
20	1	1	3	0.0	yes	yes	217.0	-	-	no	-	-	-	0.0	yes	yes	216.0	8.8
20	1	1	4	-	no	-	-	-	0.0	yes	yes	223.0	-	0.0	yes	yes	190.0	6.8
20	1	1	5	0.0	yes	yes	173.0	-	0.0	yes	yes	173.0	9.2	0.0	yes	yes	169.0	10.7
20	1	2	1	0.0	yes	yes	100.0	-	0.0	yes	yes	127.0	-	0.0	yes	yes	100.0	12.0
20	1	2	2	0.0	yes	yes	126.0	-	0.0	yes	yes	120.0	-	0.0	yes	yes	125.0	9.6
20	1	2	3	0.0	yes	yes	101.0	-	0.0	yes	yes	96.0	-	0.0	yes	yes	101.0	16.8
20	1	2	4	0.0	yes	yes	113.0	-	0.0	yes	yes	95.0	-	0.0	yes	yes	113.0	15.0
20	1	2	5	0.0	yes	yes	100.0	-	0.0	yes	yes	96.0	-	0.0	yes	yes	100.0	14.0
20	1	4	1	0.0	yes	yes	65.0	-	0.0	yes	yes	63.0	-	0.0	yes	yes	65.0	16.9
20	1	4	2	0.0	yes	yes	61.0	-	0.0	yes	yes	65.0	-	0.0	yes	yes	61.0	18.0
20	1	4	3	0.0	yes	yes	55.0	-	0.0	yes	yes	49.0	-	0.0	yes	yes	55.0	16.4
20	1	4	4	0.0	yes	yes	57.0	-	0.0	yes	yes	59.0	25.4	0.0	yes	yes	57.0	21.1
20	1	4	5	0.0	yes	yes	48.0	-	0.0	yes	yes	56.0	-	0.0	yes	yes	48.0	27.1
20	1	8	1	0.0	yes	yes	39.0	-	0.0	yes	yes	41.0	-	0.0	yes	yes	39.0	20.5
20	1	8	2	0.0	yes	yes	41.0	-	0.0	yes	yes	42.0	-	0.0	yes	yes	41.0	36.6
20	1	8	3	0.0	yes	yes	33.0	-	0.0	yes	yes	35.0	34.3	0.0	yes	yes	33.0	39.4
20	1	8	4	0.0	yes	yes	38.0	-	0.0	yes	yes	48.0	-	0.0	yes	yes	38.0	44.7
20	1	8	5	0.0	yes	yes	31.0	-	0.0	yes	yes	31.0	41.9	0.0	yes	yes	31.0	48.4
20	2	1	1	0.0	yes	yes	97.0	-	0.0	yes	yes	105.0	-	0.0	yes	yes	97.0	12.4
20	2	1	2	0.0	yes	yes	102.0	-	0.0	yes	yes	107.0	-	0.0	yes	yes	102.0	17.6
20	2	1	3	0.0	yes	yes	98.0	-	0.0	yes	yes	94.0	-	0.0	yes	yes	98.0	12.2
20	2	1	4	0.0	yes	yes	99.0	-	0.0	yes	yes	112.0	-	0.0	yes	yes	98.0	16.3
20	2	1	5	0.0	yes	yes	92.0	-	0.0	yes	yes	82.0	-	0.0	yes	yes	92.0	17.4
20	2	2	1	0.0	yes	yes	57.0	-	0.0	yes	yes	57.0	-	0.0	yes	yes	57.0	14.0
20	2	2	2	0.0	yes	yes	60.0	-	0.0	yes	yes	65.0	-	0.0	yes	yes	60.0	16.7
20	2	2	3	0.0	yes	yes	61.0	18.0	0.0	yes	yes	47.0	38.3	0.0	yes	yes	61.0	18.0
20	2	2	4	0.0	yes	yes	57.0	-	0.0	yes	yes	62.0	21.0	0.0	yes	yes	57.0	33.3
20	2	2	5	0.0	yes	yes	54.0	22.2	0.0	yes	yes	44.0	-	0.0	yes	yes	54.0	22.2
20	2	4	1	0.0	yes	yes	39.0	-	0.0	yes	yes	43.0	-	0.0	yes	yes	39.0	15.4
20	2	4	2	0.0	yes	yes	47.0	-	0.0	yes	yes	45.0	-	0.0	yes	yes	47.0	23.4
20	2	4	3	0.0	yes	yes	38.0	-	0.0	yes	yes	37.0	-	0.0	yes	yes	38.0	36.8
20	2	4	4	0.0	yes	yes	42.0	-	0.0	yes	yes	38.0	-	0.0	yes	yes	42.0	28.6
20	2	4	5	0.0	yes	yes	29.0	-	0.0	yes	yes	34.0	-	0.0	yes	yes	29.0	51.7
20	2	8	1	0.0	yes	yes	31.0	-	0.0	yes	yes	31.0	-	0.0	yes	yes	31.0	35.5
20	2	8	2	0.0	yes	yes	33.0	-	0.0	yes	yes	33.0	-	0.0	yes	yes	33.0	24.2
20	2	8	3	0.0	yes	yes	26.0	-	0.0	yes	yes	26.0	-	0.0	yes	yes	26.0	46.2
20	2	8	4	0.0	yes	yes	30.0	-	0.0	yes	yes	30.0	-	0.0	yes	yes	30.0	36.7
20	2	8	5	0.0	yes	yes	23.0	-	0.0	yes	yes	23.0	-	0.0	yes	yes	23.0	60.9
20	3	1	1	0.0	yes	yes	75.0	-	0.0	yes	yes	81.0	-	0.0	yes	yes	75.0	16.0
20	3	1	2	0.0	yes	yes	80.0	-	0.0	yes	yes	73.0	20.5	0.0	yes	yes	80.0	10.0
20	3	1	3	0.0	yes	yes	73.0	15.1	0.0	yes	yes	76.0	13.2	0.0	yes	yes	72.0	16.7
20	3	1	4	0.0	yes	yes	68.0	-	-	no	-	-	-	0.0	yes	yes	68.0	26.5
20	3	1	5	0.0	yes	yes	63.0	23.8	0.0	yes	yes	62.0	-	0.0	yes	yes	63.0	23.8
20	3	2	1	0.0	yes	yes	48.0	-	0.0	yes	yes	49.0	8.2	0.0	yes	yes	48.0	22.9
20	3	2	2	0.0	yes	yes	51.0	-	0.0	yes	yes	49.0	12.2	0.0	yes	yes	51.0	7.8
20	3	2	3	0.0	yes	yes	42.0	-	0.0	yes	yes	42.0	-	0.0	yes	yes	42.0	28.6
20	3	2	4	0.0	yes	yes	45.0	-	0.0	yes	yes	42.0	-	0.0	yes	yes	45.0	24.4
20	3	2	5	0.0	yes	yes	44.0	-	0.0	yes	yes	42.0	-	0.0	yes	yes	44.0	25.0
20	3	4	1	0.0	yes	yes	35.0	-	0.0	yes	yes	36.0	-	0.0	yes	yes	35.0	20.0
20	3	4	2	0.0	yes	yes	38.0	-	0.0	yes	yes	37.0	-	0.0	yes	yes	38.0	7.9
20	3	4	3	0.0	yes	yes	29.0	-	0.0	yes	yes	30.0	46.7	0.0	yes	yes	29.0	44.8
20	3	4	4	0.0	yes	yes	32.0	-	0.0	yes	yes	33.0	-	0.0	yes	yes	32.0	34.4
20	3	4	5	0.0	yes	yes	28.0	-	0.0	yes	yes	29.0	-	0.0	yes	yes	28.0	46.4
20	3	8	1	0.0	yes	yes	31.0	-	0.0	yes	yes	31.0	-	0.0	yes	yes	31.0	16.1
20	3	8	2	0.0	yes	yes	33.0	-	0.0	yes	yes	33.0	-	0.0	yes	yes	33.0	9.1
20	3	8	3	0.0	yes	yes	26.0	-	0.0	yes	yes	26.0	-	0.0	yes	yes	26.0	34.6
20	3	8	4	0.0	yes	yes	30.0	-	0.0	yes	yes	30.0	-	0.0	yes	yes	30.0	33.3
20	3	8	5	0.0	yes	yes	23.0	-	0.0	yes	yes	23.0	-	0.0	yes	yes	23.0	65.2

Table B.3: Problem $V, Q|sd, u, r_i, d_i|\sum C_i$ (part 1)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
40	1	1	1	0.0	yes	yes	693.0	-	0.0	yes	yes	632.0	7.9	0.0	yes	yes	692.0	3.8
40	1	1	2	0.0	yes	yes	693.0	-	0.0	yes	yes	650.0	-	0.0	yes	yes	685.0	5.0
40	1	1	3	0.0	yes	yes	757.0	-	0.0	yes	yes	671.0	-	0.0	yes	yes	753.0	7.0
40	1	1	4	0.0	yes	yes	792.0	-	0.0	yes	yes	642.0	-	0.0	yes	yes	792.0	6.2
40	1	1	5	0.0	yes	yes	759.0	-	0.0	yes	yes	581.0	7.6	0.0	yes	yes	758.0	2.6
40	1	2	1	0.0	yes	yes	373.0	-	0.0	yes	yes	343.0	-	0.0	yes	yes	370.0	5.4
40	1	2	2	0.0	yes	yes	382.0	-	0.0	yes	yes	373.0	-	0.0	yes	yes	382.0	5.5
40	1	2	3	0.0	yes	yes	346.0	-	0.0	yes	yes	419.0	-	0.0	yes	yes	346.0	10.4
40	1	2	4	0.0	yes	yes	344.0	-	0.0	yes	yes	328.0	-	0.0	yes	yes	341.0	8.5
40	1	2	5	0.0	yes	yes	392.0	-	0.0	yes	yes	347.0	-	0.0	yes	yes	392.0	6.9
40	1	4	1	0.0	yes	yes	211.0	-	0.0	yes	yes	191.0	-	0.0	yes	yes	211.0	11.8
40	1	4	2	0.0	yes	yes	214.0	-	0.0	yes	yes	175.0	-	0.0	yes	yes	214.0	14.0
40	1	4	3	0.0	yes	yes	180.0	-	0.0	yes	yes	202.0	-	0.0	yes	yes	180.0	6.1
40	1	4	4	0.0	yes	yes	168.0	-	0.0	yes	yes	161.0	8.7	0.0	yes	yes	168.0	20.2
40	1	4	5	0.0	yes	yes	185.0	-	0.0	yes	yes	177.0	-	0.0	yes	yes	185.0	11.4
40	1	8	1	0.0	yes	yes	113.0	-	0.0	yes	yes	113.0	-	0.0	yes	yes	113.0	14.2
40	1	8	2	0.0	yes	yes	119.0	-	0.0	yes	yes	113.0	-	0.0	yes	yes	119.0	15.1
40	1	8	3	0.0	yes	yes	118.0	-	0.0	yes	yes	107.0	-	0.0	yes	yes	118.0	11.9
40	1	8	4	0.0	yes	yes	107.0	-	0.0	yes	yes	113.0	-	0.0	yes	yes	107.0	23.4
40	1	8	5	0.0	yes	yes	106.0	-	0.0	yes	yes	135.0	-	0.0	yes	yes	106.0	17.0
40	2	1	1	0.0	yes	yes	378.0	-	0.0	yes	yes	308.0	-	0.1	yes	yes	376.0	5.9
40	2	1	2	0.0	yes	yes	398.0	-	0.0	yes	yes	350.0	-	0.1	yes	yes	398.0	2.5
40	2	1	3	0.0	yes	yes	372.0	-	0.0	yes	yes	359.0	-	0.1	yes	yes	365.0	8.2
40	2	1	4	0.0	yes	yes	303.0	-	0.0	yes	yes	343.0	-	0.1	yes	yes	300.0	15.7
40	2	1	5	0.0	yes	yes	316.0	-	0.0	yes	yes	337.0	-	0.1	yes	yes	316.0	13.3
40	2	2	1	0.0	yes	yes	178.0	-	0.0	yes	yes	200.0	12.5	0.0	yes	yes	178.0	10.1
40	2	2	2	0.0	yes	yes	192.0	-	0.0	yes	yes	190.0	12.6	0.1	yes	yes	191.0	11.0
40	2	2	3	0.0	yes	yes	217.0	-	0.0	yes	yes	194.0	-	0.1	yes	yes	216.0	8.8
40	2	2	4	0.0	yes	yes	202.0	-	0.0	yes	yes	192.0	-	0.0	yes	yes	201.0	19.4
40	2	2	5	0.0	yes	yes	199.0	10.1	0.0	yes	yes	177.0	13.0	0.0	yes	yes	199.0	9.5
40	2	4	1	0.0	yes	yes	102.0	-	0.0	yes	yes	116.0	-	0.0	yes	yes	102.0	18.6
40	2	4	2	0.0	yes	yes	111.0	-	0.0	yes	yes	102.0	-	0.0	yes	yes	111.0	15.3
40	2	4	3	0.0	yes	yes	118.0	-	0.0	yes	yes	115.0	-	0.0	yes	yes	118.0	23.7
40	2	4	4	0.0	yes	yes	113.0	-	0.0	yes	yes	114.0	-	0.0	yes	yes	113.0	19.5
40	2	4	5	0.0	yes	yes	101.0	-	0.0	yes	yes	114.0	-	0.0	yes	yes	101.0	20.8
40	2	8	1	0.0	yes	yes	81.0	-	0.0	yes	yes	68.0	-	0.0	yes	yes	81.0	24.7
40	2	8	2	0.0	yes	yes	75.0	-	0.0	yes	yes	80.0	-	0.0	yes	yes	75.0	26.7
40	2	8	3	0.0	yes	yes	73.0	-	0.0	yes	yes	79.0	-	0.0	yes	yes	73.0	38.4
40	2	8	4	0.0	yes	yes	69.0	-	0.0	yes	yes	67.0	-	0.0	yes	yes	69.0	37.7
40	2	8	5	0.0	yes	yes	72.0	-	0.0	yes	yes	73.0	-	0.0	yes	yes	72.0	23.6
40	3	1	1	0.0	yes	yes	212.0	-	0.0	yes	yes	229.0	-	0.1	yes	yes	212.0	11.8
40	3	1	2	0.0	yes	yes	248.0	-	0.0	yes	yes	245.0	-	0.1	yes	yes	248.0	12.1
40	3	1	3	0.0	yes	yes	248.0	-	0.0	yes	yes	249.0	10.0	0.1	yes	yes	247.0	8.9
40	3	1	4	0.0	yes	yes	222.0	-	0.0	yes	yes	231.0	-	0.1	yes	yes	222.0	12.6
40	3	1	5	0.0	yes	yes	282.0	-	0.0	yes	yes	265.0	-	0.1	yes	yes	281.0	11.0
40	3	2	1	0.0	yes	yes	154.0	-	0.0	yes	yes	146.0	-	0.0	yes	yes	154.0	11.0
40	3	2	2	0.0	yes	yes	160.0	-	0.0	yes	yes	149.0	-	0.0	yes	yes	160.0	11.9
40	3	2	3	0.0	yes	yes	153.0	-	0.0	yes	yes	144.0	-	0.0	yes	yes	153.0	18.3
40	3	2	4	0.0	yes	yes	127.0	-	0.0	yes	yes	142.0	-	0.1	yes	yes	127.0	24.4
40	3	2	5	0.0	yes	yes	130.0	-	0.0	yes	yes	148.0	-	0.0	yes	yes	130.0	13.8
40	3	4	1	0.0	yes	yes	91.0	-	0.0	yes	yes	90.0	-	0.0	yes	yes	91.0	16.5
40	3	4	2	0.0	yes	yes	87.0	-	0.0	yes	yes	94.0	-	0.0	yes	yes	87.0	18.4
40	3	4	3	0.0	yes	yes	83.0	-	0.0	yes	yes	87.0	-	0.0	yes	yes	83.0	20.5
40	3	4	4	0.0	yes	yes	82.0	-	0.0	yes	yes	83.0	-	0.0	yes	yes	82.0	29.3
40	3	4	5	0.0	yes	yes	79.0	-	0.0	yes	yes	86.0	-	0.0	yes	yes	79.0	25.3
40	3	8	1	0.0	yes	yes	70.0	-	0.0	yes	yes	65.0	-	0.0	yes	yes	70.0	28.6
40	3	8	2	0.0	yes	yes	68.0	-	0.0	yes	yes	67.0	-	0.0	yes	yes	68.0	27.9
40	3	8	3	0.0	yes	yes	65.0	-	0.0	yes	yes	64.0	-	0.0	yes	yes	65.0	29.2
40	3	8	4	0.0	yes	yes	61.0	-	0.0	yes	yes	60.0	-	0.1	yes	yes	61.0	36.1
40	3	8	5	0.0	yes	yes	63.0	-	0.0	yes	yes	61.0	-	0.0	yes	yes	63.0	25.4

Table B.4: Problem $V, Q|sd, u, r_i, d_i|\sum C_i$ (part 2)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
80	1	1	1	0.0	yes	yes	2506.0	-	0.1	yes	yes	2540.0	-	0.1	yes	yes	2506.0	3.7
80	1	1	2	0.1	yes	yes	2562.0	-	0.1	yes	yes	2305.0	-	0.2	yes	yes	2561.0	2.3
80	1	1	3	0.1	yes	yes	2625.0	3.2	0.1	yes	yes	2734.0	-	0.1	yes	yes	2625.0	3.1
80	1	1	4	0.1	yes	yes	2402.0	-	-	no	-	-	-	0.1	yes	yes	2402.0	4.2
80	1	1	5	0.1	yes	yes	2358.0	-	0.1	yes	yes	2401.0	-	0.1	yes	yes	2354.0	4.7
80	1	2	1	0.0	yes	yes	1246.0	-	0.0	yes	yes	1317.0	-	0.1	yes	yes	1245.0	4.5
80	1	2	2	0.0	yes	yes	1229.0	-	0.0	yes	yes	1324.0	-	0.1	yes	yes	1229.0	6.5
80	1	2	3	0.1	yes	yes	1377.0	-	0.0	yes	yes	1306.0	-	0.1	yes	yes	1377.0	2.8
80	1	2	4	0.0	yes	yes	1343.0	-	0.0	yes	yes	1274.0	-	0.1	yes	yes	1341.0	5.3
80	1	2	5	0.0	yes	yes	1322.0	-	0.0	yes	yes	1303.0	-	0.1	yes	yes	1322.0	5.7
80	1	4	1	0.0	yes	yes	717.0	-	0.0	yes	yes	681.0	-	0.1	yes	yes	717.0	6.6
80	1	4	2	0.0	yes	yes	735.0	-	0.0	yes	yes	654.0	-	0.1	yes	yes	735.0	5.0
80	1	4	3	0.0	yes	yes	671.0	-	0.0	yes	yes	678.0	-	0.1	yes	yes	671.0	6.7
80	1	4	4	0.0	yes	yes	722.0	-	0.0	yes	yes	685.0	8.6	0.1	yes	yes	720.0	9.7
80	1	4	5	0.0	yes	yes	656.0	7.9	0.0	yes	yes	671.0	-	0.1	yes	yes	656.0	7.9
80	1	8	1	0.0	yes	yes	391.0	-	0.0	yes	yes	375.0	-	0.1	yes	yes	391.0	14.8
80	1	8	2	0.0	yes	yes	375.0	-	0.0	yes	yes	356.0	14.9	0.0	yes	yes	375.0	13.1
80	1	8	3	0.0	yes	yes	361.0	-	0.0	yes	yes	402.0	-	0.1	yes	yes	361.0	8.0
80	1	8	4	0.0	yes	yes	356.0	-	0.0	yes	yes	355.0	-	0.1	yes	yes	356.0	16.6
80	1	8	5	0.0	yes	yes	355.0	-	0.0	yes	yes	373.0	-	0.1	yes	yes	355.0	15.5
80	2	1	1	0.1	yes	yes	1223.0	-	0.1	yes	yes	1428.0	-	0.3	yes	yes	1222.0	5.8
80	2	1	2	0.1	yes	yes	1273.0	4.3	0.1	yes	yes	1183.0	-	0.2	yes	yes	1271.0	4.7
80	2	1	3	0.0	yes	yes	1373.0	-	0.1	yes	yes	1404.0	-	0.2	yes	yes	1373.0	4.4
80	2	1	4	0.1	yes	yes	1266.0	-	0.1	yes	yes	1287.0	-	0.2	yes	yes	1266.0	6.2
80	2	1	5	0.1	yes	yes	1165.0	-	0.1	yes	yes	1295.0	-	0.2	yes	yes	1165.0	8.2
80	2	2	1	0.0	yes	yes	638.0	-	0.0	yes	yes	680.0	-	0.1	yes	yes	638.0	8.9
80	2	2	2	0.0	yes	yes	649.0	-	0.0	yes	yes	705.0	-	0.1	yes	yes	649.0	8.2
80	2	2	3	0.0	yes	yes	738.0	-	0.0	yes	yes	668.0	-	0.1	yes	yes	738.0	5.8
80	2	2	4	0.0	yes	yes	680.0	-	0.1	yes	yes	681.0	-	0.1	yes	yes	680.0	10.4
80	2	2	5	0.0	yes	yes	658.0	-	0.0	yes	yes	659.0	9.9	0.1	yes	yes	657.0	9.0
80	2	4	1	0.0	yes	yes	378.0	-	0.0	yes	yes	333.0	-	0.1	yes	yes	378.0	12.2
80	2	4	2	0.0	yes	yes	386.0	13.0	0.0	yes	yes	366.0	-	0.1	yes	yes	386.0	13.0
80	2	4	3	0.0	yes	yes	394.0	-	0.0	yes	yes	366.0	-	0.1	yes	yes	394.0	10.9
80	2	4	4	0.0	yes	yes	385.0	-	0.0	yes	yes	369.0	-	0.1	yes	yes	385.0	14.3
80	2	4	5	0.0	yes	yes	368.0	14.4	0.0	yes	yes	375.0	-	0.1	yes	yes	368.0	14.4
80	2	8	1	0.0	yes	yes	217.0	-	0.0	yes	yes	216.0	-	0.1	yes	yes	217.0	19.4
80	2	8	2	0.0	yes	yes	222.0	-	0.0	yes	yes	217.0	-	0.1	yes	yes	222.0	16.2
80	2	8	3	0.0	yes	yes	244.0	-	0.0	yes	yes	258.0	-	0.1	yes	yes	244.0	16.0
80	2	8	4	0.0	yes	yes	240.0	-	0.0	yes	yes	209.0	-	0.1	yes	yes	240.0	18.8
80	2	8	5	0.0	yes	yes	217.0	-	0.0	yes	yes	213.0	-	0.1	yes	yes	217.0	22.1
80	3	1	1	0.1	yes	yes	834.0	-	0.1	yes	yes	855.0	-	0.3	yes	yes	834.0	7.7
80	3	1	2	0.1	yes	yes	918.0	-	0.1	yes	yes	954.0	4.9	0.3	yes	yes	918.0	6.4
80	3	1	3	0.1	yes	yes	949.0	-	0.1	yes	yes	957.0	-	0.3	yes	yes	946.0	5.1
80	3	1	4	0.1	yes	yes	906.0	-	0.1	yes	yes	878.0	-	0.3	yes	yes	906.0	8.1
80	3	1	5	0.1	yes	yes	873.0	7.3	0.1	yes	yes	894.0	-	0.4	yes	yes	873.0	7.2
80	3	2	1	0.0	yes	yes	449.0	-	0.1	yes	yes	468.0	-	0.2	yes	yes	449.0	10.9
80	3	2	2	0.0	yes	yes	490.0	-	0.0	yes	yes	457.0	-	0.2	yes	yes	490.0	10.0
80	3	2	3	0.0	yes	yes	494.0	-	0.0	yes	yes	527.0	-	0.2	yes	yes	494.0	7.3
80	3	2	4	0.0	yes	yes	442.0	9.7	0.0	yes	yes	502.0	-	0.2	yes	yes	442.0	10.0
80	3	2	5	0.0	yes	yes	435.0	-	0.0	yes	yes	474.0	9.3	0.2	yes	yes	435.0	13.6
80	3	4	1	0.0	yes	yes	274.0	-	0.0	yes	yes	271.0	-	0.2	yes	yes	274.0	17.5
80	3	4	2	0.0	yes	yes	269.0	-	0.0	yes	yes	275.0	-	0.2	yes	yes	269.0	15.2
80	3	4	3	0.0	yes	yes	291.0	-	0.0	yes	yes	280.0	-	0.2	yes	yes	291.0	16.2
80	3	4	4	0.0	yes	yes	251.0	-	0.0	yes	yes	250.0	-	0.2	yes	yes	251.0	22.7
80	3	4	5	0.0	yes	yes	269.0	-	0.0	yes	yes	254.0	-	0.1	yes	yes	269.0	20.4
80	3	8	1	0.0	yes	yes	184.0	-	0.0	yes	yes	169.0	-	0.2	yes	yes	184.0	26.1
80	3	8	2	0.0	yes	yes	187.0	-	0.0	yes	yes	176.0	-	0.1	yes	yes	187.0	22.5
80	3	8	3	0.0	yes	yes	186.0	-	0.0	yes	yes	194.0	-	0.2	yes	yes	186.0	17.2
80	3	8	4	0.0	yes	yes	173.0	-	0.0	yes	yes	180.0	-	0.1	yes	yes	173.0	25.4
80	3	8	5	0.0	yes	yes	175.0	-	0.0	yes	yes	169.0	-	0.1	yes	yes	175.0	26.3

Table B.5: Problem $V, Q|sd, u, r_i, d_i|\sum C_i$ (part 3)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
160	1	1	1	0.2	yes	yes	9245.0	-	-	no	-	-	-	0.6	yes	yes	9237.0	2.5
160	1	1	2	0.3	yes	yes	10383.0	-	0.3	yes	yes	9484.0	-	0.6	yes	yes	10379.0	2.1
160	1	1	3	0.2	yes	yes	10125.0	-	0.3	yes	yes	10652.0	-	0.5	yes	yes	10124.0	1.7
160	1	1	4	0.3	yes	yes	9740.0	-	0.3	yes	yes	10046.0	-	0.6	yes	yes	9737.0	1.9
160	1	1	5	0.3	yes	yes	9954.0	-	0.4	yes	yes	10369.0	-	0.6	yes	yes	9953.0	1.5
160	1	2	1	0.1	yes	yes	4813.0	2.6	0.1	yes	yes	5208.0	-	0.4	yes	yes	4810.0	2.9
160	1	2	2	0.1	yes	yes	5073.0	-	0.1	yes	yes	5194.0	-	0.2	yes	yes	5073.0	2.3
160	1	2	3	0.1	yes	yes	4995.0	-	0.1	yes	yes	5072.0	-	0.3	yes	yes	4992.0	2.0
160	1	2	4	0.1	yes	yes	4107.0	-	0.1	yes	yes	4853.0	2.0	0.6	yes	yes	4105.0	2.5
160	1	2	5	0.1	yes	yes	5626.0	-	0.1	yes	yes	5283.0	-	0.2	yes	yes	5626.0	2.4
160	1	4	1	0.0	yes	yes	2638.0	-	0.1	yes	yes	2676.0	-	0.2	yes	yes	2638.0	4.3
160	1	4	2	0.0	yes	yes	2446.0	4.3	0.1	yes	yes	2454.0	-	0.3	yes	yes	2446.0	4.3
160	1	4	3	0.0	yes	yes	2466.0	-	0.1	yes	yes	2681.0	-	0.2	yes	yes	2466.0	3.2
160	1	4	4	0.0	yes	yes	2481.0	-	0.0	yes	yes	2546.0	-	0.3	yes	yes	2481.0	3.8
160	1	4	5	0.0	yes	yes	2576.0	-	0.1	yes	yes	2753.0	-	0.2	yes	yes	2576.0	4.0
160	1	8	1	0.0	yes	yes	1357.0	-	0.0	yes	yes	1315.0	6.8	0.2	yes	yes	1357.0	7.2
160	1	8	2	0.0	yes	yes	1413.0	-	0.0	yes	yes	1365.0	-	0.2	yes	yes	1413.0	6.2
160	1	8	3	0.0	yes	yes	1465.0	-	0.0	yes	yes	1365.0	5.1	0.2	yes	yes	1464.0	5.4
160	1	8	4	0.0	yes	yes	1350.0	-	0.0	yes	yes	1350.0	-	0.2	yes	yes	1350.0	6.7
160	1	8	5	0.0	yes	yes	1360.0	-	0.0	yes	yes	1389.0	-	0.2	yes	yes	1360.0	6.8
160	2	1	1	0.2	yes	yes	4888.0	-	0.3	yes	yes	5038.0	-	0.8	yes	yes	4888.0	2.4
160	2	1	2	0.3	yes	yes	4765.0	-	0.4	yes	yes	4969.0	-	0.9	yes	yes	4765.0	2.7
160	2	1	3	0.1	yes	yes	4928.0	-	0.3	yes	yes	5065.0	-	0.9	yes	yes	4928.0	2.1
160	2	1	4	0.2	yes	yes	5497.0	-	0.7	yes	yes	4930.0	-	0.9	yes	yes	5497.0	1.9
160	2	1	5	0.1	yes	yes	5136.0	-	0.3	yes	yes	5075.0	-	0.7	yes	yes	5136.0	2.9
160	2	2	1	0.2	yes	yes	2413.0	-	0.2	yes	yes	2830.0	-	1.0	yes	yes	2410.0	4.4
160	2	2	2	0.1	yes	yes	2515.0	-	0.2	yes	yes	2429.0	-	0.6	yes	yes	2515.0	3.3
160	2	2	3	0.1	yes	yes	2669.0	-	0.1	yes	yes	2483.0	-	0.6	yes	yes	2669.0	3.4
160	2	2	4	0.1	yes	yes	2588.0	-	0.1	yes	yes	2538.0	3.3	0.5	yes	yes	2588.0	3.2
160	2	2	5	0.1	yes	yes	2636.0	-	0.1	yes	yes	2479.0	-	0.5	yes	yes	2636.0	3.6
160	2	4	1	0.1	yes	yes	1323.0	-	0.1	yes	yes	1347.0	-	0.5	yes	yes	1323.0	7.9
160	2	4	2	0.0	yes	yes	1328.0	7.5	0.1	yes	yes	1343.0	-	0.5	yes	yes	1328.0	7.2
160	2	4	3	0.1	yes	yes	1468.0	-	0.0	yes	yes	1279.0	-	0.4	yes	yes	1468.0	5.7
160	2	4	4	0.0	yes	yes	1359.0	-	0.0	yes	yes	1296.0	-	0.5	yes	yes	1359.0	6.0
160	2	4	5	0.0	yes	yes	1317.0	-	0.0	yes	yes	1350.0	-	0.5	yes	yes	1317.0	7.4
160	2	8	1	0.0	yes	yes	791.0	-	0.0	yes	yes	719.0	-	0.5	yes	yes	791.0	10.9
160	2	8	2	0.0	yes	yes	760.0	-	0.0	yes	yes	731.0	-	0.5	yes	yes	760.0	11.7
160	2	8	3	0.0	yes	yes	718.0	-	0.0	yes	yes	722.0	-	0.4	yes	yes	718.0	8.2
160	2	8	4	0.0	yes	yes	717.0	-	0.0	yes	yes	743.0	-	0.4	yes	yes	717.0	11.4
160	2	8	5	0.0	yes	yes	729.0	-	0.0	yes	yes	744.0	-	0.5	yes	yes	729.0	12.3
160	3	1	1	0.2	yes	yes	3412.0	-	0.3	yes	yes	3325.0	-	1.6	yes	yes	3411.0	2.8
160	3	1	2	0.2	yes	yes	3308.0	-	0.3	yes	yes	3445.0	-	1.7	yes	yes	3308.0	3.9
160	3	1	3	0.2	yes	yes	3374.0	-	0.2	yes	yes	3287.0	2.7	1.4	yes	yes	3374.0	2.9
160	3	1	4	0.3	yes	yes	3473.0	-	0.4	yes	yes	3525.0	-	2.1	yes	yes	3469.0	2.5
160	3	1	5	0.2	yes	yes	3293.0	-	0.2	yes	yes	3392.0	-	1.3	yes	yes	3293.0	3.5
160	3	2	1	0.1	yes	yes	1798.0	-	0.1	yes	yes	1633.0	-	0.7	yes	yes	1798.0	5.6
160	3	2	2	0.1	yes	yes	1619.0	5.9	0.1	yes	yes	1710.0	-	1.0	yes	yes	1619.0	5.9
160	3	2	3	0.1	yes	yes	1703.0	-	0.1	yes	yes	1806.0	-	0.7	yes	yes	1703.0	4.6
160	3	2	4	0.1	yes	yes	1768.0	-	0.1	yes	yes	1742.0	4.5	0.7	yes	yes	1768.0	4.8
160	3	2	5	0.1	yes	yes	1746.0	-	0.1	yes	yes	1660.0	-	0.8	yes	yes	1746.0	5.1
160	3	4	1	0.1	yes	yes	899.0	-	0.0	yes	yes	940.0	-	0.9	yes	yes	899.0	8.9
160	3	4	2	0.0	yes	yes	1012.0	-	0.0	yes	yes	981.0	-	0.7	yes	yes	1012.0	9.1
160	3	4	3	0.0	yes	yes	953.0	-	0.0	yes	yes	936.0	8.3	0.7	yes	yes	953.0	8.1
160	3	4	4	0.0	yes	yes	969.0	-	0.0	yes	yes	884.0	-	0.7	yes	yes	969.0	8.3
160	3	4	5	0.0	yes	yes	956.0	-	0.1	yes	yes	1018.0	-	0.7	yes	yes	956.0	10.1
160	3	8	1	0.0	yes	yes	540.0	-	0.0	yes	yes	531.0	-	0.7	yes	yes	540.0	15.4
160	3	8	2	0.0	yes	yes	563.0	-	0.0	yes	yes	533.0	-	0.7	yes	yes	563.0	16.3
160	3	8	3	0.0	yes	yes	542.0	-	0.0	yes	yes	526.0	-	0.7	yes	yes	542.0	12.0
160	3	8	4	0.0	yes	yes	538.0	-	0.0	yes	yes	551.0	-	0.7	yes	yes	538.0	15.4
160	3	8	5	0.0	yes	yes	559.0	-	0.0	yes	yes	519.0	-	0.7	yes	yes	559.0	15.0

Table B.6: Problem $V, Q|sd, u, r_i, d_i|\sum C_i$ (part 4)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
20	1	1	1	0.0	yes	yes	234.0	-	0.0	yes	yes	249.0	-	0.0	yes	yes	234.0	4.3
20	1	1	2	0.0	yes	yes	265.0	-	0.0	yes	yes	273.0	-	0.0	yes	yes	265.0	6.0
20	1	1	3	0.0	yes	yes	233.0	-	0.0	yes	yes	184.0	-	0.0	yes	yes	232.0	11.6
20	1	1	4	0.0	yes	yes	233.0	9.9	0.0	yes	yes	227.0	-	0.0	yes	yes	233.0	9.4
20	1	1	5	0.0	yes	yes	236.0	7.6	0.0	yes	yes	180.0	10.0	0.0	yes	yes	236.0	7.6
20	1	2	1	0.0	yes	yes	152.0	-	0.0	yes	yes	129.0	-	0.0	yes	yes	150.0	6.7
20	1	2	2	0.0	yes	yes	149.0	4.0	0.0	yes	yes	114.0	-	0.0	yes	yes	149.0	4.0
20	1	2	3	0.0	yes	yes	121.0	-	0.0	yes	yes	132.0	-	0.0	yes	yes	121.0	9.9
20	1	2	4	0.0	yes	yes	124.0	-	0.0	yes	yes	122.0	9.0	0.0	yes	yes	123.0	17.1
20	1	2	5	0.0	yes	yes	128.0	-	0.0	yes	yes	93.0	-	0.0	yes	yes	126.0	11.1
20	1	4	1	0.0	yes	yes	74.0	-	0.0	yes	yes	72.0	-	0.0	yes	yes	74.0	10.8
20	1	4	2	0.0	yes	yes	71.0	-	0.0	yes	yes	65.0	15.4	0.0	yes	yes	71.0	12.7
20	1	4	3	0.0	yes	yes	65.0	-	0.0	yes	yes	65.0	-	0.0	yes	yes	65.0	18.5
20	1	4	4	0.0	yes	yes	81.0	-	0.0	yes	yes	78.0	-	0.0	yes	yes	81.0	19.8
20	1	4	5	0.0	yes	yes	59.0	22.0	0.0	yes	yes	67.0	-	0.0	yes	yes	59.0	22.0
20	1	8	1	0.0	yes	yes	45.0	-	0.0	yes	yes	50.0	-	0.0	yes	yes	45.0	13.3
20	1	8	2	0.0	yes	yes	49.0	-	0.0	yes	yes	50.0	-	0.0	yes	yes	49.0	18.4
20	1	8	3	0.0	yes	yes	43.0	-	0.0	yes	yes	40.0	-	0.0	yes	yes	43.0	27.9
20	1	8	4	0.0	yes	yes	45.0	-	0.0	yes	yes	45.0	-	0.0	yes	yes	45.0	17.8
20	1	8	5	0.0	yes	yes	40.0	-	0.0	yes	yes	41.0	-	0.0	yes	yes	40.0	35.0
20	2	1	1	0.0	yes	yes	132.0	-	0.0	yes	yes	101.0	9.9	0.0	yes	yes	132.0	6.8
20	2	1	2	0.0	yes	yes	131.0	-	0.0	yes	yes	121.0	-	0.0	yes	yes	131.0	9.2
20	2	1	3	0.0	yes	yes	119.0	-	0.0	yes	yes	118.0	10.2	0.0	yes	yes	119.0	10.1
20	2	1	4	0.0	yes	yes	124.0	-	0.0	yes	yes	134.0	8.2	0.0	yes	yes	124.0	12.1
20	2	1	5	0.0	yes	yes	90.0	12.2	0.0	yes	yes	125.0	12.0	0.0	yes	yes	90.0	12.2
20	2	2	1	0.0	yes	yes	69.0	15.9	0.0	yes	yes	69.0	14.5	0.0	yes	yes	69.0	15.9
20	2	2	2	0.0	yes	yes	74.0	-	0.0	yes	yes	70.0	-	0.0	yes	yes	74.0	14.9
20	2	2	3	0.0	yes	yes	65.0	-	0.0	yes	yes	59.0	27.1	0.0	yes	yes	65.0	21.5
20	2	2	4	0.0	yes	yes	67.0	-	0.0	yes	yes	64.0	17.2	0.0	yes	yes	67.0	17.9
20	2	2	5	0.0	yes	yes	73.0	-	0.0	yes	yes	59.0	-	0.0	yes	yes	73.0	15.1
20	2	4	1	0.0	yes	yes	53.0	-	0.0	yes	yes	52.0	-	0.0	yes	yes	53.0	15.1
20	2	4	2	0.0	yes	yes	58.0	-	0.0	yes	yes	51.0	-	0.0	yes	yes	58.0	15.5
20	2	4	3	0.0	yes	yes	43.0	-	0.0	yes	yes	43.0	-	0.0	yes	yes	43.0	27.9
20	2	4	4	0.0	yes	yes	45.0	-	0.0	yes	yes	50.0	-	0.0	yes	yes	45.0	22.2
20	2	4	5	0.0	yes	yes	39.0	-	0.0	yes	yes	39.0	-	0.0	yes	yes	39.0	30.8
20	2	8	1	0.0	yes	yes	38.0	-	0.0	yes	yes	34.0	-	0.0	yes	yes	38.0	13.2
20	2	8	2	0.0	yes	yes	37.0	-	0.0	yes	yes	37.0	-	0.0	yes	yes	37.0	29.7
20	2	8	3	0.0	yes	yes	29.0	-	0.0	yes	yes	32.0	-	0.0	yes	yes	29.0	41.4
20	2	8	4	0.0	yes	yes	35.0	-	0.0	yes	yes	32.0	-	0.0	yes	yes	35.0	31.4
20	2	8	5	0.0	yes	yes	28.0	-	0.0	yes	yes	25.0	-	0.0	yes	yes	28.0	35.7
20	3	1	1	0.0	yes	yes	84.0	-	0.0	yes	yes	87.0	-	0.0	yes	yes	84.0	11.9
20	3	1	2	0.0	yes	yes	84.0	-	0.0	yes	yes	95.0	-	0.0	yes	yes	84.0	10.7
20	3	1	3	0.0	yes	yes	84.0	16.7	0.0	yes	yes	94.0	-	0.0	yes	yes	84.0	15.5
20	3	1	4	0.0	yes	yes	91.0	-	0.0	yes	yes	89.0	-	0.0	yes	yes	91.0	18.7
20	3	1	5	0.0	yes	yes	76.0	-	0.0	yes	yes	85.0	17.6	0.0	yes	yes	76.0	18.4
20	3	2	1	0.0	yes	yes	55.0	-	0.0	yes	yes	57.0	15.8	0.0	yes	yes	55.0	18.2
20	3	2	2	0.0	yes	yes	59.0	-	0.0	yes	yes	59.0	-	0.0	yes	yes	59.0	5.1
20	3	2	3	0.0	yes	yes	56.0	-	0.0	yes	yes	46.0	-	0.0	yes	yes	56.0	21.4
20	3	2	4	0.0	yes	yes	52.0	-	0.0	yes	yes	56.0	-	0.0	yes	yes	52.0	21.2
20	3	2	5	0.0	yes	yes	53.0	-	0.0	yes	yes	48.0	-	0.0	yes	yes	53.0	26.4
20	3	4	1	0.0	yes	yes	41.0	-	0.0	yes	yes	38.0	-	0.0	yes	yes	41.0	17.1
20	3	4	2	0.0	yes	yes	38.0	-	0.0	yes	yes	37.0	-	0.0	yes	yes	38.0	15.8
20	3	4	3	0.0	yes	yes	31.0	-	0.0	yes	yes	32.0	-	0.0	yes	yes	31.0	38.7
20	3	4	4	0.0	yes	yes	34.0	-	0.0	yes	yes	38.0	-	0.0	yes	yes	34.0	32.4
20	3	4	5	0.0	yes	yes	34.0	-	0.0	yes	yes	32.0	-	0.0	yes	yes	34.0	35.3
20	3	8	1	0.0	yes	yes	31.0	-	0.0	yes	yes	31.0	-	0.0	yes	yes	31.0	29.0
20	3	8	2	0.0	yes	yes	33.0	-	0.0	yes	yes	33.0	-	0.0	yes	yes	33.0	27.3
20	3	8	3	0.0	yes	yes	26.0	-	0.0	yes	yes	26.0	-	0.0	yes	yes	26.0	38.5
20	3	8	4	0.0	yes	yes	30.0	-	0.0	yes	yes	30.0	-	0.0	yes	yes	30.0	33.3
20	3	8	5	0.0	yes	yes	23.0	-	0.0	yes	yes	23.0	-	0.0	yes	yes	23.0	52.2

Table B.7: Problem $V, Q|sd, u, r_i, d_i|\sum C_i$ (part 5)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
40	1	1	1	0.0	yes	yes	809.0	-	0.0	yes	yes	908.0	-	0.0	yes	yes	809.0	4.0
40	1	1	2	0.0	yes	yes	747.0	-	0.0	yes	yes	819.0	-	0.0	yes	yes	747.0	4.1
40	1	1	3	0.0	yes	yes	747.0	-	0.0	yes	yes	753.0	-	0.0	yes	yes	747.0	4.0
40	1	1	4	0.0	yes	yes	708.0	-	0.0	yes	yes	945.0	-	0.0	yes	yes	708.0	5.4
40	1	1	5	0.0	yes	yes	865.0	-	0.0	yes	yes	893.0	-	0.0	yes	yes	865.0	3.4
40	1	2	1	0.0	yes	yes	471.0	-	0.0	yes	yes	459.0	-	0.0	yes	yes	471.0	5.9
40	1	2	2	0.0	yes	yes	512.0	-	0.0	yes	yes	457.0	-	0.0	yes	yes	512.0	4.5
40	1	2	3	0.0	yes	yes	420.0	-	0.0	yes	yes	449.0	-	0.0	yes	yes	420.0	4.5
40	1	2	4	0.0	yes	yes	505.0	-	0.0	yes	yes	424.0	6.8	0.0	yes	yes	505.0	4.6
40	1	2	5	0.0	yes	yes	398.0	-	0.0	yes	yes	442.0	-	0.0	yes	yes	398.0	6.8
40	1	4	1	0.0	yes	yes	216.0	-	0.0	yes	yes	249.0	8.4	0.0	yes	yes	216.0	12.5
40	1	4	2	0.0	yes	yes	234.0	-	0.0	yes	yes	266.0	-	0.0	yes	yes	234.0	8.5
40	1	4	3	0.0	yes	yes	273.0	-	0.0	yes	yes	287.0	-	0.0	yes	yes	273.0	8.4
40	1	4	4	0.0	yes	yes	215.0	-	0.0	yes	yes	237.0	-	0.0	yes	yes	215.0	11.2
40	1	4	5	0.0	yes	yes	250.0	7.6	0.0	yes	yes	243.0	9.1	0.0	yes	yes	250.0	7.6
40	1	8	1	0.0	yes	yes	137.0	-	0.0	yes	yes	143.0	16.1	0.0	yes	yes	137.0	13.9
40	1	8	2	0.0	yes	yes	134.0	-	0.0	yes	yes	148.0	-	0.0	yes	yes	134.0	18.7
40	1	8	3	0.0	yes	yes	139.0	-	0.0	yes	yes	145.0	-	0.0	yes	yes	139.0	17.3
40	1	8	4	0.0	yes	yes	125.0	-	0.0	yes	yes	121.0	-	0.0	yes	yes	125.0	21.6
40	1	8	5	0.0	yes	yes	152.0	-	0.0	yes	yes	138.0	-	0.0	yes	yes	152.0	13.8
40	2	1	1	0.0	yes	yes	430.0	-	0.0	yes	yes	435.0	-	0.1	yes	yes	430.0	5.8
40	2	1	2	0.0	yes	yes	473.0	-	0.0	yes	yes	415.0	4.3	0.1	yes	yes	473.0	6.1
40	2	1	3	0.0	yes	yes	469.0	4.1	0.0	yes	yes	464.0	-	0.1	yes	yes	469.0	4.1
40	2	1	4	0.0	yes	yes	317.0	-	0.0	yes	yes	394.0	-	0.1	yes	yes	317.0	8.5
40	2	1	5	0.0	yes	yes	493.0	-	0.0	yes	yes	452.0	-	0.0	yes	yes	493.0	3.9
40	2	2	1	0.0	yes	yes	205.0	-	0.0	yes	yes	231.0	-	0.0	yes	yes	205.0	8.8
40	2	2	2	0.0	yes	yes	223.0	-	0.0	yes	yes	241.0	-	0.0	yes	yes	223.0	8.5
40	2	2	3	0.0	yes	yes	236.0	-	0.0	yes	yes	229.0	8.7	0.0	yes	yes	236.0	11.4
40	2	2	4	0.0	yes	yes	252.0	-	0.0	yes	yes	254.0	-	0.0	yes	yes	252.0	8.3
40	2	2	5	0.0	yes	yes	276.0	-	0.0	yes	yes	228.0	9.6	0.0	yes	yes	276.0	7.2
40	2	4	1	0.0	yes	yes	150.0	12.7	0.0	yes	yes	139.0	-	0.0	yes	yes	150.0	12.7
40	2	4	2	0.0	yes	yes	135.0	-	0.0	yes	yes	142.0	13.4	0.0	yes	yes	135.0	12.6
40	2	4	3	0.0	yes	yes	142.0	-	0.0	yes	yes	134.0	-	0.0	yes	yes	142.0	11.3
40	2	4	4	0.0	yes	yes	116.0	24.1	0.0	yes	yes	130.0	-	0.0	yes	yes	116.0	24.1
40	2	4	5	0.0	yes	yes	119.0	-	0.0	yes	yes	161.0	-	0.0	yes	yes	119.0	16.0
40	2	8	1	0.0	yes	yes	88.0	-	0.0	yes	yes	88.0	-	0.0	yes	yes	88.0	18.2
40	2	8	2	0.0	yes	yes	84.0	-	0.0	yes	yes	90.0	-	0.0	yes	yes	84.0	22.6
40	2	8	3	0.0	yes	yes	87.0	-	0.0	yes	yes	92.0	-	0.0	yes	yes	87.0	17.2
40	2	8	4	0.0	yes	yes	84.0	-	0.0	yes	yes	87.0	-	0.0	yes	yes	84.0	28.6
40	2	8	5	0.0	yes	yes	89.0	-	0.0	yes	yes	87.0	-	0.0	yes	yes	89.0	20.2
40	3	1	1	0.0	yes	yes	314.0	7.0	0.0	yes	yes	268.0	-	0.1	yes	yes	314.0	7.0
40	3	1	2	0.0	yes	yes	318.0	-	0.0	yes	yes	316.0	-	0.1	yes	yes	318.0	6.3
40	3	1	3	0.0	yes	yes	325.0	-	0.0	yes	yes	296.0	6.8	0.1	yes	yes	325.0	7.4
40	3	1	4	0.0	yes	yes	258.0	-	0.0	yes	yes	310.0	-	0.1	yes	yes	258.0	13.2
40	3	1	5	0.0	yes	yes	353.0	-	0.0	yes	yes	314.0	-	0.1	yes	yes	353.0	5.4
40	3	2	1	0.0	yes	yes	181.0	-	0.0	yes	yes	187.0	9.6	0.0	yes	yes	181.0	13.3
40	3	2	2	0.0	yes	yes	162.0	-	0.0	yes	yes	178.0	-	0.0	yes	yes	162.0	13.0
40	3	2	3	0.0	yes	yes	183.0	-	0.0	yes	yes	157.0	-	0.0	yes	yes	183.0	13.1
40	3	2	4	0.0	yes	yes	176.0	-	0.0	yes	yes	161.0	-	0.0	yes	yes	176.0	11.9
40	3	2	5	0.0	yes	yes	184.0	-	0.0	yes	yes	166.0	-	0.0	yes	yes	184.0	10.3
40	3	4	1	0.0	yes	yes	105.0	-	0.0	yes	yes	115.0	-	0.0	yes	yes	105.0	16.2
40	3	4	2	0.0	yes	yes	114.0	-	0.0	yes	yes	109.0	-	0.0	yes	yes	114.0	17.5
40	3	4	3	0.0	yes	yes	93.0	-	0.0	yes	yes	100.0	22.0	0.0	yes	yes	93.0	23.7
40	3	4	4	0.0	yes	yes	108.0	-	0.0	yes	yes	93.0	-	0.0	yes	yes	108.0	22.2
40	3	4	5	0.0	yes	yes	104.0	-	0.0	yes	yes	107.0	-	0.0	yes	yes	104.0	13.5
40	3	8	1	0.0	yes	yes	72.0	-	0.0	yes	yes	76.0	-	0.0	yes	yes	72.0	30.6
40	3	8	2	0.0	yes	yes	74.0	-	0.0	yes	yes	70.0	-	0.0	yes	yes	74.0	18.9
40	3	8	3	0.0	yes	yes	67.0	-	0.0	yes	yes	73.0	-	0.0	yes	yes	67.0	31.3
40	3	8	4	0.0	yes	yes	73.0	-	0.0	yes	yes	70.0	-	0.0	yes	yes	73.0	24.7
40	3	8	5	0.0	yes	yes	73.0	-	0.0	yes	yes	69.0	-	0.0	yes	yes	73.0	26.0

Table B.8: Problem $V, Q|sd, u, r_i, d_i|\sum C_i$ (part 6)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
80	1	1	1	0.1	yes	yes	2806.0	2.2	0.1	yes	yes	3224.0	1.7	0.1	yes	yes	2805.0	2.3
80	1	1	2	0.0	yes	yes	3237.0	-	0.0	yes	yes	3364.0	2.4	0.1	yes	yes	3237.0	1.7
80	1	1	3	0.1	yes	yes	3081.0	-	0.1	yes	yes	3220.0	2.2	0.1	yes	yes	3080.0	1.6
80	1	1	4	0.0	yes	yes	3200.0	-	0.1	yes	yes	3280.0	2.6	0.1	yes	yes	3199.0	2.9
80	1	1	5	0.0	yes	yes	3245.0	2.1	0.1	yes	yes	3296.0	-	0.1	yes	yes	3245.0	2.2
80	1	2	1	0.0	yes	yes	1732.0	-	0.0	yes	yes	1605.0	3.3	0.1	yes	yes	1732.0	2.6
80	1	2	2	0.0	yes	yes	1696.0	-	0.0	yes	yes	1667.0	3.1	0.1	yes	yes	1696.0	3.1
80	1	2	3	0.0	yes	yes	1758.0	2.6	0.0	yes	yes	1798.0	3.0	0.1	yes	yes	1758.0	2.6
80	1	2	4	0.0	yes	yes	1810.0	-	0.0	yes	yes	1613.0	4.0	0.1	yes	yes	1810.0	3.6
80	1	2	5	0.0	yes	yes	1688.0	-	0.0	yes	yes	1484.0	-	0.1	yes	yes	1687.0	2.7
80	1	4	1	0.0	yes	yes	827.0	-	0.0	yes	yes	853.0	-	0.1	yes	yes	827.0	5.2
80	1	4	2	0.0	yes	yes	942.0	-	0.0	yes	yes	910.0	-	0.1	yes	yes	942.0	4.7
80	1	4	3	0.0	yes	yes	881.0	-	0.0	yes	yes	1016.0	3.6	0.1	yes	yes	881.0	3.7
80	1	4	4	0.0	yes	yes	838.0	-	0.0	yes	yes	832.0	-	0.1	yes	yes	838.0	7.0
80	1	4	5	0.0	yes	yes	859.0	-	0.0	yes	yes	783.0	-	0.0	yes	yes	859.0	6.3
80	1	8	1	0.0	yes	yes	481.0	-	0.0	yes	yes	460.0	-	0.1	yes	yes	481.0	8.9
80	1	8	2	0.0	yes	yes	463.0	-	0.0	yes	yes	451.0	-	0.1	yes	yes	463.0	10.2
80	1	8	3	0.0	yes	yes	466.0	-	0.0	yes	yes	485.0	8.0	0.0	yes	yes	466.0	8.2
80	1	8	4	0.0	yes	yes	452.0	-	0.0	yes	yes	448.0	-	0.0	yes	yes	452.0	12.2
80	1	8	5	0.0	yes	yes	448.0	-	0.0	yes	yes	448.0	-	0.1	yes	yes	448.0	12.5
80	2	1	1	0.0	yes	yes	1760.0	-	0.1	yes	yes	1501.0	-	0.2	yes	yes	1760.0	3.1
80	2	1	2	0.0	yes	yes	1823.0	-	0.1	yes	yes	1770.0	-	0.2	yes	yes	1823.0	3.3
80	2	1	3	0.0	yes	yes	1763.0	-	0.1	yes	yes	1541.0	-	0.2	yes	yes	1763.0	2.0
80	2	1	4	0.0	yes	yes	1744.0	-	0.1	yes	yes	1722.0	3.0	0.1	yes	yes	1744.0	3.0
80	2	1	5	0.0	yes	yes	1759.0	3.2	0.1	yes	yes	1692.0	3.1	0.2	yes	yes	1759.0	3.2
80	2	2	1	0.0	yes	yes	908.0	-	0.0	yes	yes	882.0	-	0.1	yes	yes	908.0	5.4
80	2	2	2	0.0	yes	yes	914.0	-	0.0	yes	yes	847.0	5.5	0.1	yes	yes	914.0	4.5
80	2	2	3	0.0	yes	yes	936.0	-	0.0	yes	yes	796.0	-	0.1	yes	yes	936.0	5.1
80	2	2	4	0.0	yes	yes	817.0	-	0.0	yes	yes	834.0	-	0.1	yes	yes	817.0	6.2
80	2	2	5	0.0	yes	yes	886.0	6.1	0.0	yes	yes	850.0	6.2	0.1	yes	yes	886.0	6.2
80	2	4	1	0.0	yes	yes	440.0	-	0.0	yes	yes	489.0	-	0.1	yes	yes	440.0	11.1
80	2	4	2	0.0	yes	yes	421.0	-	0.0	yes	yes	468.0	-	0.1	yes	yes	421.0	11.6
80	2	4	3	0.0	yes	yes	494.0	-	0.0	yes	yes	484.0	-	0.1	yes	yes	494.0	8.3
80	2	4	4	0.0	yes	yes	415.0	-	0.0	yes	yes	446.0	-	0.1	yes	yes	415.0	15.4
80	2	4	5	0.0	yes	yes	432.0	-	0.0	yes	yes	422.0	-	0.1	yes	yes	432.0	11.6
80	2	8	1	0.0	yes	yes	257.0	-	0.0	yes	yes	241.0	19.9	0.1	yes	yes	257.0	18.3
80	2	8	2	0.0	yes	yes	271.0	-	0.0	yes	yes	267.0	-	0.1	yes	yes	271.0	17.0
80	2	8	3	0.0	yes	yes	284.0	-	0.0	yes	yes	277.0	-	0.1	yes	yes	284.0	13.4
80	2	8	4	0.0	yes	yes	271.0	-	0.0	yes	yes	265.0	-	0.1	yes	yes	271.0	19.6
80	2	8	5	0.0	yes	yes	271.0	-	0.0	yes	yes	282.0	-	0.1	yes	yes	271.0	18.8
80	3	1	1	0.0	yes	yes	1138.0	-	0.0	yes	yes	1096.0	-	0.3	yes	yes	1138.0	4.1
80	3	1	2	0.0	yes	yes	1120.0	-	0.0	yes	yes	1106.0	-	0.3	yes	yes	1120.0	4.0
80	3	1	3	0.1	yes	yes	1137.0	-	0.1	yes	yes	1256.0	4.0	0.3	yes	yes	1136.0	3.9
80	3	1	4	0.1	yes	yes	1176.0	-	0.0	yes	yes	1233.0	-	0.2	yes	yes	1176.0	4.3
80	3	1	5	0.0	yes	yes	1207.0	-	0.0	yes	yes	1100.0	5.4	0.2	yes	yes	1207.0	4.1
80	3	2	1	0.0	yes	yes	560.0	-	0.0	yes	yes	576.0	-	0.2	yes	yes	560.0	8.2
80	3	2	2	0.0	yes	yes	633.0	7.4	0.0	yes	yes	563.0	7.8	0.1	yes	yes	633.0	7.4
80	3	2	3	0.0	yes	yes	633.0	-	0.0	yes	yes	610.0	-	0.2	yes	yes	633.0	6.0
80	3	2	4	0.0	yes	yes	610.0	-	0.0	yes	yes	624.0	-	0.2	yes	yes	610.0	8.5
80	3	2	5	0.0	yes	yes	583.0	-	0.0	yes	yes	601.0	9.2	0.1	yes	yes	583.0	8.6
80	3	4	1	0.0	yes	yes	337.0	-	0.0	yes	yes	329.0	-	0.1	yes	yes	337.0	13.4
80	3	4	2	0.0	yes	yes	363.0	13.5	0.0	yes	yes	356.0	-	0.1	yes	yes	363.0	13.5
80	3	4	3	0.0	yes	yes	353.0	-	0.0	yes	yes	317.0	-	0.2	yes	yes	353.0	11.0
80	3	4	4	0.0	yes	yes	304.0	-	0.0	yes	yes	317.0	-	0.2	yes	yes	304.0	17.1
80	3	4	5	0.0	yes	yes	332.0	-	0.0	yes	yes	345.0	-	0.1	yes	yes	332.0	14.8
80	3	8	1	0.0	yes	yes	210.0	-	0.0	yes	yes	216.0	-	0.1	yes	yes	210.0	20.5
80	3	8	2	0.0	yes	yes	231.0	-	0.0	yes	yes	219.0	-	0.1	yes	yes	231.0	15.6
80	3	8	3	0.0	yes	yes	232.0	-	0.0	yes	yes	223.0	-	0.1	yes	yes	232.0	15.9
80	3	8	4	0.0	yes	yes	198.0	-	0.0	yes	yes	211.0	-	0.2	yes	yes	198.0	23.7
80	3	8	5	0.0	yes	yes	212.0	-	0.0	yes	yes	210.0	-	0.1	yes	yes	212.0	21.7

Table B.9: Problem $V, Q|sd, u, r_i, d_i|\sum C_i$ (part 7)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
160	1	1	1	0.2	yes	yes	12430.0	0.8	0.3	yes	yes	12486.0	1.0	0.5	yes	yes	12428.0	0.8
160	1	1	2	0.2	yes	yes	12610.0	-	0.2	yes	yes	12072.0	1.0	0.4	yes	yes	12610.0	0.8
160	1	1	3	0.2	yes	yes	13049.0	-	0.2	yes	yes	12959.0	-	0.4	yes	yes	13034.0	0.8
160	1	1	4	0.1	yes	yes	12800.0	-	0.3	yes	yes	12363.0	-	0.4	yes	yes	12800.0	0.8
160	1	1	5	0.1	yes	yes	14048.0	-	0.3	yes	yes	13149.0	-	0.4	yes	yes	14048.0	0.8
160	1	2	1	0.1	yes	yes	6971.0	-	0.1	yes	yes	6326.0	-	0.3	yes	yes	6971.0	1.3
160	1	2	2	0.1	yes	yes	6799.0	-	0.1	yes	yes	6201.0	-	0.3	yes	yes	6799.0	1.5
160	1	2	3	0.1	yes	yes	5672.0	-	0.1	yes	yes	6739.0	1.5	0.3	yes	yes	5672.0	1.4
160	1	2	4	0.1	yes	yes	6340.0	-	0.1	yes	yes	6660.0	-	0.2	yes	yes	6337.0	1.2
160	1	2	5	0.1	yes	yes	6505.0	-	0.1	yes	yes	6750.0	-	0.3	yes	yes	6505.0	1.5
160	1	4	1	0.0	yes	yes	3340.0	-	0.0	yes	yes	3634.0	2.3	0.2	yes	yes	3340.0	2.7
160	1	4	2	0.0	yes	yes	3258.0	-	0.1	yes	yes	3301.0	-	0.2	yes	yes	3258.0	2.6
160	1	4	3	0.1	yes	yes	3346.0	-	0.1	yes	yes	3580.0	2.0	0.2	yes	yes	3344.0	2.5
160	1	4	4	0.0	yes	yes	3168.0	-	0.0	yes	yes	3449.0	2.7	0.2	yes	yes	3168.0	2.8
160	1	4	5	0.0	yes	yes	3358.0	-	0.0	yes	yes	3489.0	-	0.2	yes	yes	3358.0	2.5
160	1	8	1	0.0	yes	yes	1743.0	-	0.0	yes	yes	1721.0	-	0.3	yes	yes	1743.0	4.7
160	1	8	2	0.0	yes	yes	1803.0	-	0.0	yes	yes	1746.0	-	0.2	yes	yes	1803.0	4.8
160	1	8	3	0.0	yes	yes	1816.0	-	0.0	yes	yes	1677.0	-	0.2	yes	yes	1816.0	3.9
160	1	8	4	0.0	yes	yes	1666.0	-	0.0	yes	yes	1775.0	-	0.2	yes	yes	1666.0	4.8
160	1	8	5	0.0	yes	yes	1717.0	-	0.0	yes	yes	1770.0	5.4	0.2	yes	yes	1717.0	6.2
160	2	1	1	0.2	yes	yes	6827.0	-	0.3	yes	yes	6601.0	-	0.8	yes	yes	6827.0	1.4
160	2	1	2	0.2	yes	yes	6706.0	-	0.2	yes	yes	6739.0	-	0.8	yes	yes	6706.0	1.8
160	2	1	3	0.2	yes	yes	6313.0	-	0.2	yes	yes	7041.0	-	0.8	yes	yes	6313.0	1.3
160	2	1	4	0.1	yes	yes	6722.0	-	0.2	yes	yes	6471.0	1.7	0.7	yes	yes	6722.0	1.7
160	2	1	5	0.2	yes	yes	6679.0	-	0.2	yes	yes	6938.0	1.4	0.7	yes	yes	6679.0	1.4
160	2	2	1	0.1	yes	yes	3323.0	2.3	0.1	yes	yes	3518.0	-	0.5	yes	yes	3323.0	2.3
160	2	2	2	0.1	yes	yes	3341.0	-	0.1	yes	yes	3181.0	2.9	0.5	yes	yes	3341.0	2.7
160	2	2	3	0.1	yes	yes	3432.0	-	0.1	yes	yes	3407.0	-	0.5	yes	yes	3432.0	2.0
160	2	2	4	0.1	yes	yes	3402.0	-	0.1	yes	yes	3424.0	-	0.6	yes	yes	3402.0	2.3
160	2	2	5	0.1	yes	yes	3399.0	-	0.1	yes	yes	3450.0	-	0.5	yes	yes	3399.0	2.7
160	2	4	1	0.0	yes	yes	1769.0	4.7	0.0	yes	yes	1847.0	-	0.4	yes	yes	1769.0	4.7
160	2	4	2	0.0	yes	yes	1857.0	-	0.0	yes	yes	1749.0	-	0.4	yes	yes	1857.0	4.9
160	2	4	3	0.0	yes	yes	1700.0	-	0.0	yes	yes	1810.0	-	0.4	yes	yes	1700.0	4.2
160	2	4	4	0.0	yes	yes	1739.0	-	0.0	yes	yes	1666.0	-	0.4	yes	yes	1739.0	4.3
160	2	4	5	0.0	yes	yes	1724.0	-	0.0	yes	yes	1810.0	-	0.4	yes	yes	1724.0	5.5
160	2	8	1	0.0	yes	yes	980.0	-	0.0	yes	yes	955.0	-	0.4	yes	yes	980.0	9.6
160	2	8	2	0.0	yes	yes	915.0	-	0.0	yes	yes	906.0	-	0.4	yes	yes	915.0	10.1
160	2	8	3	0.0	yes	yes	925.0	-	0.0	yes	yes	943.0	-	0.4	yes	yes	925.0	7.7
160	2	8	4	0.0	yes	yes	933.0	-	0.0	yes	yes	928.0	-	0.4	yes	yes	933.0	9.5
160	2	8	5	0.0	yes	yes	933.0	-	0.0	yes	yes	897.0	10.9	0.4	yes	yes	933.0	9.9
160	3	1	1	0.2	yes	yes	4428.0	-	0.2	yes	yes	4362.0	2.0	1.4	yes	yes	4428.0	1.9
160	3	1	2	0.1	yes	yes	4268.0	-	0.2	yes	yes	4340.0	-	1.1	yes	yes	4268.0	2.3
160	3	1	3	0.1	yes	yes	4248.0	-	0.2	yes	yes	4414.0	1.7	1.0	yes	yes	4247.0	1.7
160	3	1	4	0.2	yes	yes	4017.0	-	0.2	yes	yes	4256.0	-	1.3	yes	yes	4017.0	2.4
160	3	1	5	0.2	yes	yes	4410.0	-	0.2	yes	yes	4320.0	-	1.2	yes	yes	4410.0	1.7
160	3	2	1	0.1	yes	yes	2174.0	-	0.1	yes	yes	2285.0	3.5	0.7	yes	yes	2174.0	3.7
160	3	2	2	0.1	yes	yes	2303.0	-	0.1	yes	yes	2390.0	-	1.0	yes	yes	2303.0	4.4
160	3	2	3	0.1	yes	yes	2368.0	-	0.1	yes	yes	2307.0	-	0.7	yes	yes	2368.0	3.3
160	3	2	4	0.1	yes	yes	2211.0	-	0.1	yes	yes	2464.0	3.1	0.7	yes	yes	2211.0	3.4
160	3	2	5	0.1	yes	yes	2338.0	-	0.1	yes	yes	2438.0	-	0.7	yes	yes	2338.0	3.5
160	3	4	1	0.0	yes	yes	1249.0	-	0.0	yes	yes	1246.0	-	0.7	yes	yes	1249.0	6.9
160	3	4	2	0.0	yes	yes	1225.0	-	0.0	yes	yes	1165.0	-	0.7	yes	yes	1225.0	7.7
160	3	4	3	0.0	yes	yes	1212.0	-	0.0	yes	yes	1153.0	-	0.7	yes	yes	1212.0	6.5
160	3	4	4	0.0	yes	yes	1176.0	-	0.0	yes	yes	1237.0	-	0.7	yes	yes	1176.0	7.1
160	3	4	5	0.0	yes	yes	1300.0	-	0.0	yes	yes	1261.0	-	0.7	yes	yes	1300.0	7.0
160	3	8	1	0.0	yes	yes	738.0	-	0.0	yes	yes	694.0	-	0.7	yes	yes	738.0	10.3
160	3	8	2	0.0	yes	yes	731.0	-	0.0	yes	yes	648.0	-	0.7	yes	yes	731.0	11.1
160	3	8	3	0.0	yes	yes	687.0	-	0.0	yes	yes	673.0	-	0.7	yes	yes	687.0	9.6
160	3	8	4	0.0	yes	yes	692.0	-	0.0	yes	yes	679.0	-	0.7	yes	yes	692.0	11.6
160	3	8	5	0.0	yes	yes	647.0	-	0.0	yes	yes	700.0	-	0.7	yes	yes	647.0	14.7

Table B.10: Problem $V, Q|sd, u, r_i, d_i|\sum C_i$ (part 8)

n	V	Q	instance	CPU(s)	$\sum C_i$	opt	gapGr(%)
20	1	1	1	0.0	117.0	yes	6.8
20	1	1	2	0.0	140.0	yes	0.0
20	1	1	3	0.0	106.0	yes	2.8
20	1	1	4	0.0	110.0	yes	13.6
20	1	1	5	0.0	99.0	yes	8.1
20	1	2	1	0.1	89.0	yes	10.1
20	1	2	2	0.1	99.0	yes	9.1
20	1	2	3	0.1	83.0	yes	3.6
20	1	2	4	0.1	91.0	yes	7.7
20	1	2	5	0.0	68.0	yes	10.3
20	1	4	1	0.1	80.0	yes	22.5
20	1	4	2	0.1	85.0	yes	12.9
20	1	4	3	0.1	79.0	yes	12.7
20	1	4	4	0.1	67.0	yes	16.4
20	1	4	5	0.0	56.0	yes	23.2
20	1	8	1	0.1	69.0	yes	14.5
20	1	8	2	0.1	79.0	yes	13.9
20	1	8	3	0.1	67.0	yes	9.0
20	1	8	4	0.2	73.0	yes	19.2
20	1	8	5	0.0	57.0	yes	21.1
20	2	1	1	0.0	70.0	yes	5.7
20	2	1	2	0.0	82.0	yes	0.0
20	2	1	3	0.0	60.0	yes	5.0
20	2	1	4	0.0	65.0	yes	9.2
20	2	1	5	0.0	56.0	yes	7.1
20	2	2	1	0.0	53.0	yes	7.5
20	2	2	2	0.1	70.0	yes	2.9
20	2	2	3	0.0	50.0	yes	8.0
20	2	2	4	0.1	54.0	yes	11.1
20	2	2	5	0.0	42.0	yes	14.3
20	2	4	1	0.1	54.0	yes	13.0
20	2	4	2	0.0	55.0	yes	14.5
20	2	4	3	0.1	47.0	yes	4.3
20	2	4	4	0.1	49.0	yes	8.2
20	2	4	5	0.1	46.0	yes	10.9
20	2	8	1	0.0	44.0	yes	15.9
20	2	8	2	0.1	51.0	yes	5.9
20	2	8	3	0.1	45.0	yes	13.3
20	2	8	4	0.1	44.0	yes	9.1
20	2	8	5	0.0	39.0	yes	10.3
20	3	1	1	0.0	54.0	yes	3.7
20	3	1	2	0.0	63.0	yes	0.0
20	3	1	3	0.0	46.0	yes	6.5
20	3	1	4	0.0	50.0	yes	6.0
20	3	1	5	0.0	42.0	yes	7.1
20	3	2	1	0.1	47.0	yes	4.3
20	3	2	2	0.0	52.0	yes	7.7
20	3	2	3	0.0	36.0	yes	13.9
20	3	2	4	0.0	43.0	yes	2.3
20	3	2	5	0.0	33.0	yes	18.2
20	3	4	1	0.0	39.0	yes	2.6
20	3	4	2	0.0	48.0	yes	8.3
20	3	4	3	0.0	35.0	yes	22.9
20	3	4	4	0.0	37.0	yes	0.0
20	3	4	5	0.0	32.0	yes	15.6
20	3	8	1	0.0	38.0	yes	5.3
20	3	8	2	0.0	42.0	yes	9.5
20	3	8	3	0.0	30.0	yes	16.7
20	3	8	4	0.1	43.0	yes	4.7
20	3	8	5	0.0	29.0	yes	10.3

n	V	Q	instance	CPU(s)	$\sum C_i$	opt	gapGr(%)
40	1	1	1	0.1	346.0	yes	4.9
40	1	1	2	0.1	379.0	yes	4.0
40	1	1	3	0.1	476.0	yes	6.7
40	1	1	4	0.1	419.0	yes	3.3
40	1	1	5	0.1	400.0	yes	9.3
40	1	2	1	0.9	272.0	yes	7.0
40	1	2	2	0.6	262.0	yes	11.5
40	1	2	3	0.3	307.0	yes	11.1
40	1	2	4	1.2	297.0	yes	5.7
40	1	2	5	0.2	278.0	yes	20.5
40	1	4	1	4.2	216.0	yes	15.3
40	1	4	2	0.9	228.0	yes	4.8
40	1	4	3	2.5	249.0	yes	19.3
40	1	4	4	1.7	265.0	yes	4.9
40	1	4	5	0.4	290.0	yes	7.6
40	1	8	1	8.9	199.0	yes	7.5
40	1	8	2	2.0	213.0	yes	19.2
40	1	8	3	2.1	258.0	yes	22.5
40	1	8	4	3.7	206.0	yes	15.0
40	1	8	5	2.2	169.0	yes	9.5
40	2	1	1	0.1	191.0	yes	7.9
40	2	1	2	0.1	201.0	yes	3.0
40	2	1	3	0.0	255.0	yes	5.9
40	2	1	4	0.0	222.0	yes	3.6
40	2	1	5	0.1	212.0	yes	6.6
40	2	2	1	0.5	155.0	yes	11.6
40	2	2	2	0.3	153.0	yes	9.8
40	2	2	3	0.1	165.0	yes	16.4
40	2	2	4	2.8	149.0	yes	4.7
40	2	2	5	0.3	163.0	yes	8.0
40	2	4	1	1.2	125.0	yes	8.8
40	2	4	2	0.5	119.0	yes	8.4
40	2	4	3	1.2	173.0	yes	17.3
40	2	4	4	0.5	133.0	yes	7.5
40	2	4	5	0.6	122.0	yes	13.9
40	2	8	1	7.6	147.0	yes	8.8
40	2	8	2	1.4	129.0	yes	12.4
40	2	8	3	3.8	150.0	yes	13.3
40	2	8	4	0.4	130.0	yes	13.1
40	2	8	5	2.6	119.0	yes	8.4
40	3	1	1	0.2	141.0	yes	7.1
40	3	1	2	0.1	145.0	yes	2.8
40	3	1	3	0.1	181.0	yes	8.3
40	3	1	4	0.1	157.0	yes	3.2
40	3	1	5	0.1	150.0	yes	4.7
40	3	2	1	0.4	112.0	yes	10.7
40	3	2	2	0.2	114.0	yes	8.8
40	3	2	3	0.3	141.0	yes	9.9
40	3	2	4	0.5	120.0	yes	4.2
40	3	2	5	0.2	115.0	yes	8.7
40	3	4	1	0.3	103.0	yes	17.5
40	3	4	2	1.1	98.0	yes	8.2
40	3	4	3	23.9	128.0	yes	11.7
40	3	4	4	2.1	101.0	yes	10.9
40	3	4	5	2.0	95.0	yes	8.4
40	3	8	1	1.3	97.0	yes	17.5
40	3	8	2	0.5	82.0	yes	1.2
40	3	8	3	0.5	100.0	yes	19.0
40	3	8	4	0.4	87.0	yes	10.3
40	3	8	5	2.4	87.0	yes	13.8

Table B.11: Problem $V, Q|sd|\sum C_i$ (part 1)

n	V	Q	instance	CPU(s)	$\sum C_i$	opt	gapGr(%)
80	1	1	1	0.4	1345.0	yes	3.4
80	1	1	2	0.5	1304.0	yes	2.1
80	1	1	3	0.4	1453.0	yes	6.5
80	1	1	4	0.4	1559.0	yes	7.4
80	1	1	5	0.3	1360.0	yes	4.7
80	1	2	1	1802.6	960.0	no	5.6
80	1	2	2	2.5	945.0	yes	9.2
80	1	2	3	91.6	978.0	yes	13.7
80	1	2	4	14.2	1067.0	yes	8.8
80	1	2	5	638.8	961.0	yes	8.4
80	1	4	1	1800.1	792.0	no	12.5
80	1	4	2	1801.6	752.0	no	10.4
80	1	4	3	1801.4	799.0	no	18.0
80	1	4	4	41.8	988.0	yes	12.8
80	1	4	5	343.4	816.0	yes	8.8
80	1	8	1	1801.6	680.0	no	14.4
80	1	8	2	1801.2	680.0	no	12.9
80	1	8	3	1800.3	777.0	no	8.4
80	1	8	4	1801.5	832.0	no	10.2
80	1	8	5	1801.8	643.0	no	13.7
80	2	1	1	0.5	703.0	yes	2.8
80	2	1	2	0.4	681.0	yes	2.2
80	2	1	3	0.4	767.0	yes	6.5
80	2	1	4	0.4	809.0	yes	7.2
80	2	1	5	0.4	706.0	yes	5.1
80	2	2	1	1801.5	518.0	no	6.9
80	2	2	2	51.3	483.0	yes	7.0
80	2	2	3	1802.3	553.0	no	12.7
80	2	2	4	6.4	572.0	yes	12.9
80	2	2	5	1804.0	533.0	no	5.1
80	2	4	1	1801.0	436.0	no	12.6
80	2	4	2	1800.1	406.0	no	11.6
80	2	4	3	1800.2	459.0	no	10.7
80	2	4	4	1016.0	517.0	yes	9.9
80	2	4	5	312.2	428.0	yes	10.0
80	2	8	1	1801.1	367.0	no	15.3
80	2	8	2	1800.2	419.0	no	15.5
80	2	8	3	1801.8	385.0	no	16.9
80	2	8	4	1801.4	420.0	no	20.7
80	2	8	5	1801.2	387.0	no	14.2
80	3	1	1	0.5	489.0	yes	2.7
80	3	1	2	0.4	475.0	yes	2.3
80	3	1	3	0.4	539.0	yes	6.5
80	3	1	4	0.3	559.0	yes	7.5
80	3	1	5	0.3	488.0	yes	5.5
80	3	2	1	45.9	390.0	yes	5.9
80	3	2	2	57.3	345.0	yes	8.4
80	3	2	3	5.8	358.0	yes	9.5
80	3	2	4	1806.0	376.0	no	11.2
80	3	2	5	9.3	354.0	yes	13.3
80	3	4	1	84.3	281.0	yes	13.2
80	3	4	2	683.3	287.0	yes	13.9
80	3	4	3	90.6	327.0	yes	9.5
80	3	4	4	1800.8	347.0	no	16.4
80	3	4	5	95.4	320.0	yes	12.8
80	3	8	1	1801.6	301.0	no	11.3
80	3	8	2	1801.7	252.0	no	11.1
80	3	8	3	1801.6	331.0	no	9.1
80	3	8	4	1801.4	320.0	no	12.2
80	3	8	5	1800.8	273.0	no	15.0

n	V	Q	instance	CPU(s)	$\sum C_i$	opt	gapGr(%)
160	1	1	1	7.6	5271.0	yes	1.2
160	1	1	2	2.5	5246.0	yes	1.3
160	1	1	3	2.3	4736.0	yes	0.6
160	1	1	4	2.0	4912.0	yes	2.8
160	1	1	5	1.6	4906.0	yes	1.3
160	1	2	1	1801.6	3806.0	no	4.5
160	1	2	2	1801.4	3552.0	no	7.0
160	1	2	3	1801.3	3306.0	no	5.0
160	1	2	4	1800.3	3314.0	no	10.0
160	1	2	5	1800.2	3370.0	no	8.7
160	1	4	1	1800.2	2918.0	no	6.7
160	1	4	2	1800.4	3067.0	no	4.8
160	1	4	3	1800.3	2838.0	no	3.9
160	1	4	4	1800.2	2858.0	no	6.3
160	1	4	5	1801.4	2668.0	no	6.4
160	1	8	1	1800.4	2925.0	no	4.2
160	1	8	2	1800.3	2583.0	no	6.1
160	1	8	3	1800.3	2261.0	no	4.2
160	1	8	4	1800.4	2525.0	no	5.4
160	1	8	5	1800.1	2612.0	no	10.8
160	2	1	1	3.1	2688.0	yes	1.4
160	2	1	2	6.2	2676.0	yes	1.2
160	2	1	3	4.0	2433.0	yes	0.6
160	2	1	4	1.9	2517.0	yes	2.5
160	2	1	5	1.8	2511.0	yes	1.1
160	2	2	1	1801.5	1879.0	no	3.7
160	2	2	2	1801.6	1840.0	no	6.2
160	2	2	3	1801.4	1748.0	no	6.6
160	2	2	4	1801.8	1751.0	no	5.6
160	2	2	5	33.7	1760.0	yes	9.9
160	2	4	1	1801.4	1517.0	no	4.4
160	2	4	2	1800.2	1589.0	no	7.4
160	2	4	3	1800.1	1446.0	no	5.7
160	2	4	4	1800.1	1367.0	no	8.1
160	2	4	5	1800.3	1482.0	no	7.5
160	2	8	1	1800.3	1616.0	no	4.9
160	2	8	2	1800.3	1304.0	no	7.2
160	2	8	3	1800.2	1273.0	no	3.3
160	2	8	4	1800.5	1309.0	no	5.8
160	2	8	5	1801.2	1098.0	no	8.1
160	3	1	1	2.8	1828.0	yes	1.6
160	3	1	2	2.3	1820.0	yes	1.3
160	3	1	3	3.1	1666.0	yes	0.8
160	3	1	4	2.6	1721.0	yes	2.3
160	3	1	5	1.8	1713.0	yes	1.1
160	3	2	1	1800.3	1316.0	no	6.2
160	3	2	2	1801.7	1356.0	no	4.6
160	3	2	3	1801.5	1214.0	no	5.1
160	3	2	4	1801.3	1158.0	no	5.4
160	3	2	5	1802.1	1178.0	no	7.8
160	3	4	1	1800.3	1138.0	no	8.3
160	3	4	2	1801.2	1121.0	no	3.7
160	3	4	3	1800.3	1070.0	no	5.6
160	3	4	4	1800.4	1016.0	no	6.7
160	3	4	5	1801.2	1037.0	no	10.8
160	3	8	1	1800.3	987.0	no	5.7
160	3	8	2	1800.4	1036.0	no	6.2
160	3	8	3	1800.3	970.0	no	4.8
160	3	8	4	1802.7	935.0	no	10.1
160	3	8	5	1800.2	980.0	no	5.6

Table B.12: Problem $V, Q|sd|\sum C_i$ (part 2)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
20	1	1	1	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	140.0	6.4
20	1	1	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	177.0	9.6
20	1	1	3	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	122.0	24.6
20	1	1	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	140.0	12.1
20	1	1	5	-	no	-	-	-	0.0	yes	yes	142.0	-	0.0	yes	yes	119.0	14.3
20	1	2	1	-	no	-	-	-	0.0	yes	yes	118.0	-	0.0	yes	yes	137.0	10.2
20	1	2	2	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	123.0	26.0
20	1	2	3	-	no	-	-	-	0.0	yes	yes	123.0	-	0.0	yes	yes	116.0	21.6
20	1	2	4	-	no	-	-	-	0.0	yes	yes	131.0	-	0.0	yes	yes	122.0	15.6
20	1	2	5	-	no	-	-	-	0.0	yes	yes	104.0	-	0.0	yes	yes	109.0	22.0
20	1	4	1	-	no	-	-	-	0.0	yes	yes	112.0	-	0.0	yes	yes	142.0	16.2
20	1	4	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	135.0	8.9
20	1	4	3	-	no	-	-	-	0.0	yes	yes	105.0	-	0.0	yes	yes	109.0	19.3
20	1	4	4	-	no	-	-	-	0.0	yes	yes	126.0	-	0.0	yes	yes	129.0	16.3
20	1	4	5	0.0	yes	yes	91.0	14.3	0.0	yes	yes	105.0	-	0.0	yes	yes	89.0	16.9
20	1	8	1	0.0	yes	yes	109.0	-	0.0	yes	yes	123.0	-	0.0	yes	yes	109.0	14.7
20	1	8	2	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	118.0	7.6
20	1	8	3	-	no	-	-	-	0.0	yes	yes	100.0	-	0.0	yes	yes	113.0	15.0
20	1	8	4	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	132.0	17.4
20	1	8	5	0.0	yes	yes	110.0	-	0.0	yes	yes	106.0	17.0	0.0	yes	yes	107.0	21.5
20	2	1	1	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	75.0	9.3
20	2	1	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	94.0	22.3
20	2	1	3	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	71.0	28.2
20	2	1	4	-	no	-	-	-	0.0	yes	yes	79.0	-	0.0	yes	yes	83.0	19.3
20	2	1	5	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	64.0	23.4
20	2	2	1	-	no	-	-	-	0.0	yes	yes	79.0	-	0.0	yes	yes	66.0	12.1
20	2	2	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	81.0	17.3
20	2	2	3	-	no	-	-	-	0.0	yes	yes	64.0	-	0.0	yes	yes	60.0	18.3
20	2	2	4	0.0	yes	yes	68.0	-	0.0	yes	yes	72.0	-	0.0	yes	yes	68.0	20.6
20	2	2	5	-	no	-	-	-	0.0	yes	yes	53.0	-	0.0	yes	yes	51.0	49.0
20	2	4	1	0.0	yes	yes	67.0	-	-	no	-	-	-	0.0	yes	yes	67.0	22.4
20	2	4	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	70.0	20.0
20	2	4	3	0.0	yes	yes	74.0	-	0.0	yes	yes	55.0	-	0.0	yes	yes	71.0	21.1
20	2	4	4	-	no	-	-	-	0.0	yes	yes	68.0	-	0.1	yes	yes	60.0	28.3
20	2	4	5	-	no	-	-	-	0.0	yes	yes	58.0	-	0.0	yes	yes	63.0	28.6
20	2	8	1	0.0	yes	yes	69.0	-	0.0	yes	yes	64.0	-	0.0	yes	yes	69.0	13.0
20	2	8	2	-	no	-	-	-	0.0	yes	yes	74.0	-	0.0	yes	yes	67.0	20.9
20	2	8	3	0.0	yes	yes	62.0	-	0.0	yes	yes	63.0	-	0.0	yes	yes	62.0	29.0
20	2	8	4	0.0	yes	yes	71.0	-	0.0	yes	yes	57.0	-	0.0	yes	yes	68.0	26.5
20	2	8	5	0.0	yes	yes	59.0	-	0.0	yes	yes	62.0	-	0.0	yes	yes	57.0	33.3
20	3	1	1	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	62.0	11.3
20	3	1	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	68.0	8.8
20	3	1	3	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	47.0	44.7
20	3	1	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	63.0	17.5
20	3	1	5	-	no	-	-	-	0.0	yes	yes	51.0	-	0.0	yes	yes	50.0	22.0
20	3	2	1	-	no	-	-	-	0.0	yes	yes	54.0	-	0.0	yes	yes	50.0	18.0
20	3	2	2	0.0	yes	yes	62.0	-	0.0	yes	yes	58.0	-	0.0	yes	yes	58.0	8.6
20	3	2	3	0.0	yes	yes	44.0	-	0.0	yes	yes	45.0	-	0.0	yes	yes	44.0	29.5
20	3	2	4	-	no	-	-	-	0.0	yes	yes	48.0	-	0.1	yes	yes	56.0	23.2
20	3	2	5	0.0	yes	yes	48.0	-	0.0	yes	yes	41.0	-	0.0	yes	yes	46.0	34.8
20	3	4	1	-	no	-	-	-	0.0	yes	yes	47.0	-	0.0	yes	yes	45.0	22.2
20	3	4	2	-	no	-	-	-	0.0	yes	yes	56.0	-	0.1	yes	yes	55.0	21.8
20	3	4	3	0.0	yes	yes	46.0	-	0.0	yes	yes	45.0	-	0.0	yes	yes	45.0	26.7
20	3	4	4	0.0	yes	yes	49.0	-	0.0	yes	yes	49.0	-	0.0	yes	yes	49.0	24.5
20	3	4	5	-	no	-	-	-	0.0	yes	yes	40.0	-	0.0	yes	yes	36.0	55.6
20	3	8	1	0.0	yes	yes	46.0	-	0.0	yes	yes	54.0	-	0.0	yes	yes	46.0	13.0
20	3	8	2	0.0	yes	yes	52.0	-	0.0	yes	yes	63.0	-	0.0	yes	yes	52.0	11.5
20	3	8	3	0.0	yes	yes	46.0	-	0.0	yes	yes	44.0	29.5	0.0	yes	yes	46.0	21.7
20	3	8	4	-	no	-	-	-	0.0	yes	yes	50.0	-	0.0	yes	yes	51.0	31.4
20	3	8	5	0.0	yes	yes	42.0	-	0.0	yes	yes	41.0	-	0.0	yes	yes	42.0	28.6

Table B.13: Problem $V, Q|sd, r_i, d_i|\sum C_i$ (part 1)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
40	1	1	1	-	no	-	-	-	0.0	yes	yes	542.0	-	0.0	yes	yes	492.0	17.5
40	1	1	2	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	484.0	11.4
40	1	1	3	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	530.0	8.7
40	1	1	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	525.0	4.8
40	1	1	5	-	no	-	-	-	-	no	-	-	-	0.2	yes	yes	535.0	8.4
40	1	2	1	-	no	-	-	-	-	no	-	-	-	0.2	yes	yes	418.0	11.0
40	1	2	2	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	429.0	15.9
40	1	2	3	0.1	yes	yes	433.0	-	0.1	yes	yes	417.0	-	0.6	yes	yes	421.0	6.7
40	1	2	4	-	no	-	-	-	0.0	yes	yes	418.0	-	0.2	yes	yes	381.0	9.7
40	1	2	5	-	no	-	-	-	0.1	yes	yes	413.0	-	0.3	yes	yes	398.0	13.3
40	1	4	1	0.0	yes	yes	386.0	-	0.1	yes	yes	346.0	-	0.1	yes	yes	385.0	9.4
40	1	4	2	0.1	yes	yes	389.0	-	-	no	-	-	-	0.1	yes	yes	378.0	9.5
40	1	4	3	0.1	yes	yes	387.0	-	0.1	yes	yes	413.0	-	0.7	yes	yes	366.0	14.5
40	1	4	4	-	no	-	-	-	0.1	yes	yes	367.0	-	0.1	yes	yes	424.0	18.2
40	1	4	5	-	no	-	-	-	-	no	-	-	-	0.2	yes	yes	404.0	16.3
40	1	8	1	0.0	yes	yes	390.0	-	0.1	yes	yes	379.0	-	0.1	yes	yes	388.0	9.8
40	1	8	2	0.0	yes	yes	381.0	-	0.1	yes	yes	400.0	-	0.2	yes	yes	379.0	16.9
40	1	8	3	-	no	-	-	-	-	no	-	-	-	0.6	yes	yes	451.0	14.6
40	1	8	4	0.0	yes	yes	385.0	-	0.2	yes	yes	382.0	-	0.1	yes	yes	385.0	9.1
40	1	8	5	0.0	yes	yes	371.0	-	0.0	yes	yes	354.0	-	0.1	yes	yes	361.0	7.8
40	2	1	1	-	no	-	-	-	0.1	yes	yes	261.0	-	0.1	yes	yes	264.0	11.4
40	2	1	2	-	no	-	-	-	0.0	yes	yes	259.0	-	0.1	yes	yes	278.0	14.7
40	2	1	3	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	293.0	11.9
40	2	1	4	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	253.0	4.0
40	2	1	5	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	249.0	15.3
40	2	2	1	-	no	-	-	-	0.1	yes	yes	218.0	-	0.3	yes	yes	237.0	10.1
40	2	2	2	-	no	-	-	-	0.1	yes	yes	220.0	-	0.3	yes	yes	219.0	9.6
40	2	2	3	-	no	-	-	-	0.1	yes	yes	223.0	-	0.3	yes	yes	229.0	16.2
40	2	2	4	0.1	yes	yes	195.0	-	0.1	yes	yes	209.0	-	0.1	yes	yes	193.0	23.3
40	2	2	5	-	no	-	-	-	-	no	-	-	-	0.2	yes	yes	212.0	20.3
40	2	4	1	0.1	yes	yes	205.0	-	0.1	yes	yes	220.0	-	0.2	yes	yes	188.0	16.0
40	2	4	2	0.1	yes	yes	192.0	-	0.1	yes	yes	201.0	-	0.5	yes	yes	191.0	14.1
40	2	4	3	-	no	-	-	-	0.2	yes	yes	227.0	-	0.7	yes	yes	225.0	10.2
40	2	4	4	0.1	yes	yes	214.0	-	0.1	yes	yes	221.0	-	0.1	yes	yes	211.0	16.1
40	2	4	5	0.1	yes	yes	195.0	-	0.1	yes	yes	204.0	-	0.3	yes	yes	189.0	14.8
40	2	8	1	0.2	yes	yes	223.0	-	0.2	yes	yes	217.0	-	0.4	yes	yes	216.0	14.8
40	2	8	2	0.1	yes	yes	229.0	-	0.1	yes	yes	225.0	-	0.2	yes	yes	227.0	7.9
40	2	8	3	0.1	yes	yes	234.0	-	0.1	yes	yes	235.0	-	0.2	yes	yes	230.0	17.0
40	2	8	4	-	no	-	-	-	0.1	yes	yes	204.0	-	0.3	yes	yes	240.0	13.3
40	2	8	5	0.1	yes	yes	220.0	-	0.1	yes	yes	207.0	-	0.1	yes	yes	217.0	16.1
40	3	1	1	-	no	-	-	-	0.0	yes	yes	180.0	-	0.1	yes	yes	176.0	17.0
40	3	1	2	-	no	-	-	-	0.0	yes	yes	182.0	-	0.1	yes	yes	182.0	12.6
40	3	1	3	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	208.0	6.7
40	3	1	4	-	no	-	-	-	0.0	yes	yes	185.0	-	0.1	yes	yes	202.0	20.3
40	3	1	5	-	no	-	-	-	0.0	yes	yes	194.0	-	0.1	yes	yes	177.0	9.0
40	3	2	1	0.2	yes	yes	163.0	-	0.2	yes	yes	139.0	-	0.3	yes	yes	159.0	13.8
40	3	2	2	-	no	-	-	-	0.2	yes	yes	154.0	-	0.7	yes	yes	154.0	15.6
40	3	2	3	-	no	-	-	-	0.0	yes	yes	196.0	-	0.5	yes	yes	167.0	13.2
40	3	2	4	0.0	yes	yes	163.0	-	0.0	yes	yes	151.0	-	0.2	yes	yes	158.0	15.2
40	3	2	5	0.1	yes	yes	151.0	-	-	no	-	-	-	0.2	yes	yes	144.0	25.7
40	3	4	1	0.0	yes	yes	143.0	11.9	0.1	yes	yes	154.0	-	0.2	yes	yes	142.0	14.1
40	3	4	2	0.0	yes	yes	146.0	-	0.2	yes	yes	145.0	-	0.2	yes	yes	144.0	17.4
40	3	4	3	-	no	-	-	-	-	no	-	-	-	0.4	yes	yes	172.0	13.4
40	3	4	4	0.1	yes	yes	148.0	-	0.1	yes	yes	154.0	-	0.2	yes	yes	145.0	24.8
40	3	4	5	0.0	yes	yes	142.0	-	0.0	yes	yes	176.0	-	0.1	yes	yes	141.0	22.7
40	3	8	1	0.1	yes	yes	157.0	-	0.0	yes	yes	162.0	-	0.1	yes	yes	157.0	14.0
40	3	8	2	0.1	yes	yes	172.0	-	0.0	yes	yes	147.0	-	0.2	yes	yes	172.0	15.7
40	3	8	3	-	no	-	-	-	0.2	yes	yes	140.0	-	0.7	yes	yes	144.0	14.6
40	3	8	4	0.1	yes	yes	132.0	-	0.1	yes	yes	135.0	-	0.3	yes	yes	132.0	23.5
40	3	8	5	0.1	yes	yes	149.0	-	0.0	yes	yes	139.0	-	0.2	yes	yes	147.0	12.9

Table B.14: Problem $V, Q|sd, r_i, d_i|\sum C_i$ (part 2)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
80	1	1	1	-	no	-	-	-	0.2	yes	yes	1754.0	-	2.2	yes	yes	1782.0	6.9
80	1	1	2	-	no	-	-	-	0.1	yes	yes	1833.0	-	0.3	yes	yes	1763.0	8.0
80	1	1	3	-	no	-	-	-	0.2	yes	yes	1886.0	-	0.3	yes	yes	1815.0	7.2
80	1	1	4	-	no	-	-	-	-	no	-	-	-	0.3	yes	yes	1744.0	10.8
80	1	1	5	-	no	-	-	-	0.3	yes	yes	1781.0	-	0.4	yes	yes	1695.0	6.9
80	1	2	1	1.6	yes	yes	1610.0	-	9.5	yes	yes	1488.0	-	2.9	yes	yes	1558.0	10.3
80	1	2	2	1.9	yes	yes	1691.0	-	16.5	yes	yes	1432.0	-	17.6	yes	yes	1665.0	6.6
80	1	2	3	-	no	-	-	-	17.1	yes	yes	1591.0	-	6.2	yes	yes	1395.0	10.3
80	1	2	4	-	no	-	-	-	11.9	yes	yes	1502.0	-	1.5	yes	yes	1378.0	13.1
80	1	2	5	0.5	yes	yes	1567.0	-	1.0	yes	yes	1487.0	-	3.0	yes	yes	1519.0	13.8
80	1	4	1	0.9	yes	yes	1426.0	-	4.0	yes	yes	1415.0	-	5.2	yes	yes	1393.0	10.5
80	1	4	2	0.9	yes	yes	1433.0	-	2.6	yes	yes	1523.0	6.8	1.3	yes	yes	1427.0	9.6
80	1	4	3	0.8	yes	yes	1462.0	-	6.8	yes	yes	1411.0	-	10.3	yes	yes	1446.0	6.2
80	1	4	4	-	no	-	-	-	5.0	yes	yes	1559.0	-	5.2	yes	yes	1471.0	9.9
80	1	4	5	-	no	-	-	-	1.4	yes	yes	1506.0	-	2.7	yes	yes	1364.0	9.5
80	1	8	1	0.4	yes	yes	1207.0	-	0.8	yes	yes	1490.0	-	0.7	yes	yes	1197.0	8.0
80	1	8	2	0.2	yes	yes	1327.0	-	0.5	yes	yes	1400.0	-	0.6	yes	yes	1325.0	6.9
80	1	8	3	0.8	yes	yes	1436.0	-	2.7	yes	yes	1411.0	-	11.9	yes	yes	1416.0	11.4
80	1	8	4	0.8	yes	yes	1448.0	-	1.5	yes	yes	1404.0	-	2.4	yes	yes	1434.0	10.3
80	1	8	5	0.4	yes	yes	1326.0	-	0.3	yes	yes	1352.0	-	0.6	yes	yes	1323.0	7.0
80	2	1	1	-	no	-	-	-	-	no	-	-	-	1.7	yes	yes	943.0	9.0
80	2	1	2	-	no	-	-	-	-	no	-	-	-	1.1	yes	yes	821.0	12.1
80	2	1	3	-	no	-	-	-	0.2	yes	yes	1016.0	-	1.1	yes	yes	900.0	7.8
80	2	1	4	-	no	-	-	-	0.2	yes	yes	926.0	-	0.6	yes	yes	921.0	8.8
80	2	1	5	-	no	-	-	-	0.1	yes	yes	872.0	-	0.5	yes	yes	880.0	10.0
80	2	2	1	3.5	yes	yes	765.0	-	4.6	yes	yes	757.0	-	89.4	yes	yes	748.0	13.0
80	2	2	2	0.7	yes	yes	788.0	-	1.2	yes	yes	738.0	-	2.5	yes	yes	775.0	10.7
80	2	2	3	1.6	yes	yes	814.0	-	1.7	yes	yes	784.0	-	4.9	yes	yes	801.0	14.1
80	2	2	4	0.8	yes	yes	833.0	-	2.0	yes	yes	804.0	-	2.5	yes	yes	802.0	12.8
80	2	2	5	0.6	yes	yes	767.0	-	1.0	yes	yes	747.0	-	2.9	yes	yes	752.0	11.3
80	2	4	1	6.1	yes	yes	695.0	-	1.3	yes	yes	759.0	-	14.2	yes	yes	683.0	13.5
80	2	4	2	4.6	yes	yes	680.0	-	99.2	yes	yes	631.0	-	14.5	yes	yes	680.0	13.5
80	2	4	3	6.5	yes	yes	802.0	-	16.7	yes	yes	713.0	-	10.7	yes	yes	794.0	10.3
80	2	4	4	18.3	yes	yes	730.0	-	16.6	yes	yes	765.0	-	15.9	yes	yes	711.0	11.8
80	2	4	5	1.9	yes	yes	708.0	-	1.0	yes	yes	759.0	-	1.2	yes	yes	702.0	13.4
80	2	8	1	0.5	yes	yes	776.0	-	1.6	yes	yes	826.0	-	1.6	yes	yes	775.0	8.3
80	2	8	2	1.0	yes	yes	807.0	-	2.3	yes	yes	717.0	-	2.0	yes	yes	801.0	13.4
80	2	8	3	0.6	yes	yes	712.0	-	0.9	yes	yes	649.0	-	2.7	yes	yes	709.0	12.4
80	2	8	4	2.5	yes	yes	701.0	-	2.7	yes	yes	690.0	-	3.7	yes	yes	695.0	14.8
80	2	8	5	1.1	yes	yes	729.0	-	0.8	yes	yes	762.0	-	1.8	yes	yes	715.0	11.2
80	3	1	1	-	no	-	-	-	0.4	yes	yes	613.0	-	0.8	yes	yes	594.0	13.8
80	3	1	2	-	no	-	-	-	0.1	yes	yes	642.0	-	1.0	yes	yes	585.0	8.7
80	3	1	3	-	no	-	-	-	-	no	-	-	-	1.0	yes	yes	651.0	8.8
80	3	1	4	-	no	-	-	-	0.3	yes	yes	675.0	-	0.8	yes	yes	627.0	12.1
80	3	1	5	-	no	-	-	-	0.2	yes	yes	617.0	-	0.7	yes	yes	598.0	11.4
80	3	2	1	12.6	yes	yes	540.0	-	2.2	yes	yes	541.0	-	20.6	yes	yes	515.0	13.8
80	3	2	2	0.9	yes	yes	593.0	-	2.9	yes	yes	525.0	-	2.4	yes	yes	582.0	12.4
80	3	2	3	0.7	yes	yes	538.0	-	0.9	yes	yes	521.0	-	3.5	yes	yes	534.0	10.5
80	3	2	4	0.5	yes	yes	580.0	-	1.0	yes	yes	514.0	-	4.2	yes	yes	560.0	16.6
80	3	2	5	2.1	yes	yes	504.0	-	1.0	yes	yes	505.0	-	4.0	yes	yes	497.0	10.9
80	3	4	1	4.3	yes	yes	508.0	-	7.7	yes	yes	475.0	-	55.1	yes	yes	508.0	11.0
80	3	4	2	0.7	yes	yes	508.0	-	5.0	yes	yes	510.0	-	3.8	yes	yes	507.0	15.2
80	3	4	3	0.9	yes	yes	501.0	-	8.9	yes	yes	482.0	-	2.5	yes	yes	500.0	11.8
80	3	4	4	1.9	yes	yes	507.0	-	8.3	yes	yes	519.0	-	12.2	yes	yes	493.0	14.4
80	3	4	5	1.6	yes	yes	482.0	-	6.1	yes	yes	537.0	-	3.3	yes	yes	481.0	16.4
80	3	8	1	2.0	yes	yes	503.0	-	3.1	yes	yes	527.0	-	2.0	yes	yes	500.0	9.0
80	3	8	2	0.3	yes	yes	527.0	9.7	0.1	yes	yes	503.0	-	0.8	yes	yes	526.0	10.1
80	3	8	3	2.0	yes	yes	523.0	-	1.6	yes	yes	532.0	-	6.2	yes	yes	515.0	14.4
80	3	8	4	1.1	yes	yes	457.0	-	10.8	yes	yes	484.0	-	8.2	yes	yes	452.0	16.6
80	3	8	5	0.2	yes	yes	497.0	-	0.3	yes	yes	518.0	-	0.7	yes	yes	497.0	13.5

Table B.15: Problem $V, Q|sd, r_i, d_i|\sum C_i$ (part 3)

n	V	Q	instance	t-t					t-w					t				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
160	1	1	1	-	no	-	-	-	1.8	yes	yes	7002.0	-	12.5	yes	yes	6562.0	7.4
160	1	1	2	-	no	-	-	-	3.6	yes	yes	6872.0	-	4.3	yes	yes	6995.0	8.0
160	1	1	3	-	no	-	-	-	2.9	yes	yes	6482.0	-	5.1	yes	yes	6363.0	5.2
160	1	1	4	-	no	-	-	-	3.5	yes	yes	6399.0	-	18.0	yes	yes	6477.0	5.3
160	1	1	5	-	no	-	-	-	5.7	yes	yes	6388.0	-	2.8	yes	yes	6370.0	5.1
160	1	2	1	1800.8	yes	no	6037.0	-	1800.1	yes	no	5715.0	-	1800.3	yes	no	5830.0	9.0
160	1	2	2	93.4	yes	yes	5495.0	-	19.1	yes	yes	5689.0	-	103.1	yes	yes	5444.0	7.7
160	1	2	3	418.7	yes	yes	5339.0	-	232.6	yes	yes	5477.0	-	1800.2	yes	no	5321.0	6.8
160	1	2	4	222.4	yes	yes	5522.0	-	355.1	yes	yes	5699.0	-	230.0	yes	yes	5456.0	7.4
160	1	2	5	158.6	yes	yes	5580.0	-	392.0	yes	yes	5323.0	-	1207.9	yes	yes	5550.0	7.5
160	1	4	1	1018.3	yes	yes	5117.0	-	1801.3	yes	no	5071.0	-	1801.6	yes	no	5089.0	3.9
160	1	4	2	198.7	yes	yes	5277.0	-	188.6	yes	yes	5157.0	-	274.1	yes	yes	5239.0	8.0
160	1	4	3	24.6	yes	yes	5482.0	-	217.2	yes	yes	5151.0	-	51.2	yes	yes	5471.0	5.3
160	1	4	4	506.3	yes	yes	5010.0	-	305.9	yes	yes	5640.0	-	1801.5	yes	no	4991.0	6.7
160	1	4	5	8.4	yes	yes	5535.0	-	9.9	yes	yes	5047.0	-	5.6	yes	yes	5513.0	4.2
160	1	8	1	218.7	yes	yes	5393.0	-	596.6	yes	yes	4949.0	-	40.4	yes	yes	5365.0	7.6
160	1	8	2	17.9	yes	yes	5062.0	-	9.3	yes	yes	5080.0	-	18.7	yes	yes	5051.0	5.7
160	1	8	3	4.5	yes	yes	5167.0	-	4.9	yes	yes	5116.0	-	4.2	yes	yes	5161.0	3.2
160	1	8	4	64.5	yes	yes	5001.0	-	37.2	yes	yes	5482.0	-	105.5	yes	yes	4983.0	8.0
160	1	8	5	14.8	yes	yes	5087.0	-	9.5	yes	yes	5159.0	-	19.4	yes	yes	5076.0	4.4
160	2	1	1	-	no	-	-	-	2.0	yes	yes	3443.0	-	33.6	yes	yes	3280.0	7.8
160	2	1	2	-	no	-	-	-	4.2	yes	yes	3410.0	-	5.7	yes	yes	3225.0	6.6
160	2	1	3	-	no	-	-	-	2.5	yes	yes	3139.0	-	16.2	yes	yes	3095.0	6.3
160	2	1	4	-	no	-	-	-	4.0	yes	yes	3297.0	-	14.7	yes	yes	3114.0	6.0
160	2	1	5	-	no	-	-	-	1.7	yes	yes	3210.0	-	4.1	yes	yes	3171.0	4.1
160	2	2	1	1800.9	yes	no	2939.0	-	1800.8	yes	no	2686.0	-	1801.2	yes	no	2930.0	7.8
160	2	2	2	1800.1	yes	no	3022.0	-	1800.1	yes	no	2649.0	-	1681.9	yes	yes	2970.0	8.9
160	2	2	3	1800.1	yes	no	2841.0	-	570.7	yes	yes	2709.0	-	1800.4	yes	no	2825.0	8.1
160	2	2	4	1800.1	yes	no	2580.0	-	258.8	yes	yes	2828.0	-	1801.4	yes	no	2557.0	8.5
160	2	2	5	57.9	yes	yes	2708.0	-	344.5	yes	yes	3056.0	-	117.7	yes	yes	2687.0	7.3
160	2	4	1	1800.7	yes	no	2688.0	-	482.2	yes	yes	2607.0	-	1801.2	yes	no	2682.0	7.4
160	2	4	2	1010.1	yes	yes	2712.0	-	1801.6	yes	no	2933.0	-	1801.8	yes	no	2704.0	10.1
160	2	4	3	101.8	yes	yes	2669.0	-	110.4	yes	yes	2758.0	-	1800.5	yes	no	2669.0	5.0
160	2	4	4	21.9	yes	yes	2819.0	-	531.7	yes	yes	2562.0	-	22.1	yes	yes	2815.0	4.9
160	2	4	5	79.2	yes	yes	2681.0	-	19.2	yes	yes	2667.0	-	49.1	yes	yes	2664.0	9.0
160	2	8	1	1800.9	yes	no	2925.0	-	45.6	yes	yes	2764.0	-	1801.1	yes	no	2919.0	7.8
160	2	8	2	6.4	yes	yes	2639.0	-	2.7	yes	yes	2794.0	-	10.3	yes	yes	2638.0	5.7
160	2	8	3	11.3	yes	yes	2549.0	-	365.8	yes	yes	2585.0	-	24.4	yes	yes	2535.0	7.3
160	2	8	4	8.2	yes	yes	2834.0	-	153.2	yes	yes	2630.0	-	15.3	yes	yes	2812.0	4.8
160	2	8	5	2.4	yes	yes	2822.0	-	3.0	yes	yes	2752.0	-	3.2	yes	yes	2819.0	4.6
160	3	1	1	-	no	-	-	-	4.1	yes	yes	2336.0	-	21.0	yes	yes	2312.0	7.5
160	3	1	2	-	no	-	-	-	9.3	yes	yes	2261.0	-	10.5	yes	yes	2188.0	9.5
160	3	1	3	-	no	-	-	-	0.9	yes	yes	2192.0	-	6.3	yes	yes	2180.0	7.2
160	3	1	4	-	no	-	-	-	4.6	yes	yes	2156.0	-	4.6	yes	yes	2196.0	9.3
160	3	1	5	-	no	-	-	-	2.2	yes	yes	2263.0	-	3.7	yes	yes	2248.0	9.4
160	3	2	1	1800.2	yes	no	1972.0	-	1800.8	yes	no	1969.0	-	374.5	yes	yes	1940.0	11.2
160	3	2	2	1800.1	yes	no	2003.0	-	132.0	yes	yes	1950.0	-	1800.6	yes	no	1983.0	10.0
160	3	2	3	1800.1	yes	no	1795.0	-	168.9	yes	yes	1872.0	-	1800.5	yes	no	1772.0	10.3
160	3	2	4	485.3	yes	yes	1874.0	-	1800.2	yes	no	1881.0	-	716.0	yes	yes	1868.0	7.8
160	3	2	5	1800.3	yes	no	1802.0	-	49.7	yes	yes	1835.0	-	215.9	yes	yes	1794.0	9.9
160	3	4	1	133.5	yes	yes	1982.0	-	1102.8	yes	yes	1867.0	-	69.0	yes	yes	1976.0	8.0
160	3	4	2	1046.9	yes	yes	1737.0	-	1801.0	yes	no	1875.0	-	1614.1	yes	yes	1724.0	12.2
160	3	4	3	23.9	yes	yes	1858.0	-	281.6	yes	yes	1823.0	-	24.9	yes	yes	1852.0	8.8
160	3	4	4	1244.5	yes	yes	1891.0	-	1801.4	yes	no	1925.0	-	845.7	yes	yes	1889.0	7.7
160	3	4	5	3.2	yes	yes	1923.0	-	1800.2	yes	no	1775.0	-	11.7	yes	yes	1921.0	8.2
160	3	8	1	1801.2	yes	no	1688.0	-	1801.3	yes	no	1834.0	-	1801.4	yes	no	1686.0	10.9
160	3	8	2	198.0	yes	yes	1842.0	-	535.0	yes	yes	1909.0	-	199.2	yes	yes	1837.0	10.3
160	3	8	3	34.0	yes	yes	1773.0	-	1.4	yes	yes	1905.0	-	29.8	yes	yes	1768.0	9.6
160	3	8	4	4.4	yes	yes	1809.0	-	1801.1	yes	no	1684.0	-	11.1	yes	yes	1808.0	6.3
160	3	8	5	29.1	yes	yes	1829.0	-	450.7	yes	yes	1691.0	-	25.3	yes	yes	1825.0	10.1

Table B.16: Problem $V, Q|sd, r_i, d_i|\sum C_i$ (part 4)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
20	1	1	1	-	no	-	-	-	0.0	yes	yes	148.0	-	0.0	yes	yes	157.0	21.7
20	1	1	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	154.0	16.2
20	1	1	3	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	143.0	18.9
20	1	1	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	153.0	12.4
20	1	1	5	-	no	-	-	-	0.0	yes	yes	135.0	-	0.0	yes	yes	120.0	24.2
20	1	2	1	0.0	yes	yes	149.0	-	0.0	yes	yes	144.0	-	0.0	yes	yes	149.0	12.1
20	1	2	2	-	no	-	-	-	0.0	yes	yes	156.0	-	0.0	yes	yes	146.0	15.1
20	1	2	3	-	no	-	-	-	0.0	yes	yes	144.0	-	0.0	yes	yes	162.0	18.5
20	1	2	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	120.0	15.8
20	1	2	5	0.0	yes	yes	142.0	9.9	0.0	yes	yes	135.0	-	0.0	yes	yes	142.0	9.9
20	1	4	1	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	146.0	9.6
20	1	4	2	-	no	-	-	-	0.0	yes	yes	161.0	-	0.0	yes	yes	153.0	12.4
20	1	4	3	0.0	yes	yes	122.0	-	-	no	-	-	-	0.0	yes	yes	120.0	12.5
20	1	4	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	127.0	9.4
20	1	4	5	0.0	yes	yes	108.0	-	-	no	-	-	-	0.0	yes	yes	106.0	18.9
20	1	8	1	-	no	-	-	-	0.0	yes	yes	124.0	-	0.0	yes	yes	127.0	7.1
20	1	8	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	144.0	9.7
20	1	8	3	0.0	yes	yes	127.0	-	0.0	yes	yes	109.0	16.5	0.0	yes	yes	127.0	13.4
20	1	8	4	-	no	-	-	-	0.0	yes	yes	150.0	-	0.0	yes	yes	154.0	7.8
20	1	8	5	0.0	yes	yes	116.0	-	0.0	yes	yes	132.0	-	0.0	yes	yes	115.0	14.8
20	2	1	1	-	no	-	-	-	0.0	yes	yes	84.0	-	0.0	yes	yes	95.0	8.4
20	2	1	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	106.0	8.5
20	2	1	3	-	no	-	-	-	0.0	yes	yes	74.0	-	0.0	yes	yes	91.0	15.4
20	2	1	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	100.0	8.0
20	2	1	5	-	no	-	-	-	0.0	yes	yes	74.0	-	0.0	yes	yes	75.0	16.0
20	2	2	1	0.0	yes	yes	71.0	-	0.0	yes	yes	77.0	-	0.0	yes	yes	71.0	23.9
20	2	2	2	0.0	yes	yes	89.0	-	-	no	-	-	-	0.0	yes	yes	86.0	10.5
20	2	2	3	0.0	yes	yes	68.0	-	0.0	yes	yes	71.0	22.5	0.0	yes	yes	68.0	20.6
20	2	2	4	0.0	yes	yes	83.0	-	0.0	yes	yes	85.0	-	0.0	yes	yes	83.0	18.1
20	2	2	5	-	no	-	-	-	0.0	yes	yes	68.0	-	0.0	yes	yes	64.0	28.1
20	2	4	1	-	no	-	-	-	0.0	yes	yes	65.0	-	0.0	yes	yes	101.0	13.9
20	2	4	2	0.0	yes	yes	79.0	-	0.0	yes	yes	97.0	-	0.0	yes	yes	78.0	20.5
20	2	4	3	0.0	yes	yes	68.0	-	0.0	yes	yes	76.0	11.8	0.0	yes	yes	63.0	23.8
20	2	4	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	84.0	16.7
20	2	4	5	-	no	-	-	-	0.0	yes	yes	76.0	-	0.0	yes	yes	67.0	13.4
20	2	8	1	0.0	yes	yes	84.0	-	0.0	yes	yes	78.0	-	0.0	yes	yes	84.0	6.0
20	2	8	2	0.0	yes	yes	79.0	-	0.0	yes	yes	79.0	-	0.0	yes	yes	78.0	9.0
20	2	8	3	0.0	yes	yes	67.0	-	0.0	yes	yes	65.0	-	0.0	yes	yes	66.0	18.2
20	2	8	4	0.0	yes	yes	76.0	-	0.0	yes	yes	70.0	-	0.0	yes	yes	75.0	20.0
20	2	8	5	0.0	yes	yes	71.0	-	0.0	yes	yes	62.0	-	0.0	yes	yes	71.0	22.5
20	3	1	1	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	64.0	32.8
20	3	1	2	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	72.0	5.6
20	3	1	3	0.0	yes	yes	53.0	-	-	no	-	-	-	0.0	yes	yes	52.0	25.0
20	3	1	4	-	no	-	-	-	0.0	yes	yes	72.0	-	0.0	yes	yes	63.0	17.5
20	3	1	5	-	no	-	-	-	0.0	yes	yes	54.0	-	0.0	yes	yes	64.0	15.6
20	3	2	1	-	no	-	-	-	0.0	yes	yes	59.0	-	0.1	yes	yes	58.0	8.6
20	3	2	2	-	no	-	-	-	0.0	yes	yes	65.0	-	0.0	yes	yes	65.0	4.6
20	3	2	3	0.0	yes	yes	50.0	-	0.0	yes	yes	49.0	-	0.0	yes	yes	49.0	22.4
20	3	2	4	0.0	yes	yes	60.0	-	0.0	yes	yes	54.0	-	0.0	yes	yes	60.0	26.7
20	3	2	5	0.0	yes	yes	47.0	-	0.0	yes	yes	49.0	-	0.0	yes	yes	46.0	28.3
20	3	4	1	0.0	yes	yes	57.0	-	0.0	yes	yes	54.0	7.4	0.0	yes	yes	57.0	17.5
20	3	4	2	-	no	-	-	-	0.0	yes	yes	58.0	-	0.1	yes	yes	64.0	15.6
20	3	4	3	0.0	yes	yes	47.0	-	0.0	yes	yes	53.0	28.3	0.0	yes	yes	47.0	25.5
20	3	4	4	0.0	yes	yes	61.0	-	0.0	yes	yes	47.0	-	0.0	yes	yes	61.0	18.0
20	3	4	5	0.0	yes	yes	42.0	-	0.0	yes	yes	45.0	35.6	0.0	yes	yes	42.0	42.9
20	3	8	1	0.0	yes	yes	54.0	-	0.0	yes	yes	57.0	-	0.0	yes	yes	54.0	13.0
20	3	8	2	0.0	yes	yes	52.0	-	0.0	yes	yes	49.0	-	0.0	yes	yes	52.0	13.5
20	3	8	3	0.0	yes	yes	52.0	15.4	0.0	yes	yes	52.0	-	0.0	yes	yes	51.0	17.6
20	3	8	4	0.0	yes	yes	57.0	-	0.0	yes	yes	61.0	-	0.0	yes	yes	57.0	31.6
20	3	8	5	0.0	yes	yes	50.0	-	0.0	yes	yes	50.0	-	0.0	yes	yes	50.0	30.0

Table B.17: Problem $V, Q|sd, r_i, d_i|\sum C_i$ (part 5)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
40	1	1	1	-	no	-	-	-	0.0	yes	yes	579.0	-	0.1	yes	yes	528.0	3.6
40	1	1	2	0.0	yes	yes	503.0	-	0.0	yes	yes	548.0	-	0.1	yes	yes	483.0	15.3
40	1	1	3	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	633.0	8.4
40	1	1	4	-	no	-	-	-	-	no	-	-	-	0.0	yes	yes	535.0	8.4
40	1	1	5	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	585.0	7.0
40	1	2	1	0.0	yes	yes	472.0	-	0.1	yes	yes	444.0	-	0.1	yes	yes	471.0	6.2
40	1	2	2	0.0	yes	yes	459.0	-	0.0	yes	yes	483.0	-	0.1	yes	yes	441.0	6.3
40	1	2	3	0.0	yes	yes	474.0	-	0.1	yes	yes	497.0	-	0.3	yes	yes	456.0	7.2
40	1	2	4	0.0	yes	yes	477.0	-	0.0	yes	yes	490.0	-	0.1	yes	yes	472.0	8.7
40	1	2	5	-	no	-	-	-	0.0	yes	yes	496.0	-	0.1	yes	yes	451.0	13.5
40	1	4	1	-	no	-	-	-	0.0	yes	yes	459.0	-	0.1	yes	yes	429.0	9.3
40	1	4	2	0.0	yes	yes	519.0	-	0.0	yes	yes	449.0	-	0.1	yes	yes	504.0	5.8
40	1	4	3	0.0	yes	yes	497.0	-	0.0	yes	yes	470.0	-	0.1	yes	yes	486.0	7.6
40	1	4	4	0.0	yes	yes	450.0	-	0.0	yes	yes	465.0	-	0.1	yes	yes	450.0	7.8
40	1	4	5	0.0	yes	yes	533.0	-	0.1	yes	yes	589.0	-	0.1	yes	yes	518.0	6.9
40	1	8	1	0.0	yes	yes	397.0	-	0.0	yes	yes	441.0	-	0.1	yes	yes	393.0	9.7
40	1	8	2	0.0	yes	yes	435.0	-	0.0	yes	yes	448.0	-	0.1	yes	yes	431.0	10.9
40	1	8	3	0.0	yes	yes	455.0	-	0.1	yes	yes	516.0	-	0.1	yes	yes	450.0	12.2
40	1	8	4	0.0	yes	yes	384.0	-	0.0	yes	yes	459.0	-	0.0	yes	yes	384.0	10.4
40	1	8	5	0.0	yes	yes	433.0	-	0.0	yes	yes	484.0	-	0.1	yes	yes	432.0	5.6
40	2	1	1	0.0	yes	yes	261.0	-	0.0	yes	yes	298.0	-	0.1	yes	yes	255.0	10.6
40	2	1	2	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	294.0	7.1
40	2	1	3	-	no	-	-	-	-	no	-	-	-	0.1	yes	yes	318.0	10.4
40	2	1	4	-	no	-	-	-	0.0	yes	yes	285.0	-	0.0	yes	yes	291.0	10.7
40	2	1	5	-	no	-	-	-	0.0	yes	yes	285.0	-	0.1	yes	yes	286.0	11.9
40	2	2	1	0.0	yes	yes	269.0	-	0.1	yes	yes	284.0	-	0.1	yes	yes	263.0	12.9
40	2	2	2	-	no	-	-	-	0.0	yes	yes	268.0	-	0.1	yes	yes	256.0	15.2
40	2	2	3	0.1	yes	yes	268.0	-	0.1	yes	yes	248.0	-	0.2	yes	yes	265.0	12.5
40	2	2	4	0.0	yes	yes	234.0	-	0.0	yes	yes	235.0	-	0.1	yes	yes	228.0	12.7
40	2	2	5	0.0	yes	yes	263.0	-	0.1	yes	yes	229.0	-	0.1	yes	yes	261.0	10.3
40	2	4	1	0.0	yes	yes	235.0	-	0.0	yes	yes	271.0	-	0.1	yes	yes	235.0	17.9
40	2	4	2	-	no	-	-	-	0.1	yes	yes	269.0	-	0.1	yes	yes	258.0	8.9
40	2	4	3	0.1	yes	yes	265.0	-	0.1	yes	yes	231.0	-	0.2	yes	yes	263.0	14.8
40	2	4	4	0.0	yes	yes	231.0	-	0.0	yes	yes	277.0	-	0.1	yes	yes	231.0	12.1
40	2	4	5	0.0	yes	yes	267.0	-	0.1	yes	yes	263.0	-	0.1	yes	yes	267.0	10.5
40	2	8	1	0.0	yes	yes	252.0	-	0.0	yes	yes	245.0	-	0.1	yes	yes	251.0	11.6
40	2	8	2	0.0	yes	yes	244.0	-	0.0	yes	yes	263.0	-	0.1	yes	yes	240.0	11.3
40	2	8	3	-	no	-	-	-	0.2	yes	yes	251.0	-	0.2	yes	yes	234.0	13.7
40	2	8	4	0.0	yes	yes	262.0	-	0.1	yes	yes	242.0	-	0.1	yes	yes	259.0	15.8
40	2	8	5	0.0	yes	yes	222.0	10.8	0.0	yes	yes	239.0	-	0.1	yes	yes	222.0	14.0
40	3	1	1	0.0	yes	yes	198.0	-	0.0	yes	yes	197.0	-	0.1	yes	yes	187.0	8.0
40	3	1	2	-	no	-	-	-	0.0	yes	yes	213.0	-	0.1	yes	yes	195.0	10.3
40	3	1	3	-	no	-	-	-	0.0	yes	yes	218.0	-	0.1	yes	yes	213.0	13.1
40	3	1	4	-	no	-	-	-	0.0	yes	yes	188.0	-	0.1	yes	yes	199.0	7.5
40	3	1	5	-	no	-	-	-	0.0	yes	yes	212.0	-	0.1	yes	yes	201.0	10.0
40	3	2	1	0.0	yes	yes	201.0	-	0.0	yes	yes	189.0	-	0.1	yes	yes	199.0	9.0
40	3	2	2	0.0	yes	yes	188.0	-	0.0	yes	yes	169.0	-	0.1	yes	yes	184.0	13.0
40	3	2	3	0.0	yes	yes	191.0	-	0.2	yes	yes	190.0	-	0.2	yes	yes	185.0	13.5
40	3	2	4	0.0	yes	yes	183.0	-	0.0	yes	yes	204.0	-	0.2	yes	yes	180.0	16.7
40	3	2	5	0.0	yes	yes	203.0	-	0.0	yes	yes	183.0	-	0.1	yes	yes	203.0	11.8
40	3	4	1	0.0	yes	yes	189.0	-	0.0	yes	yes	192.0	-	0.2	yes	yes	188.0	13.3
40	3	4	2	0.0	yes	yes	167.0	-	0.0	yes	yes	178.0	-	0.1	yes	yes	167.0	13.2
40	3	4	3	0.1	yes	yes	193.0	-	0.1	yes	yes	173.0	-	0.3	yes	yes	192.0	21.4
40	3	4	4	0.1	yes	yes	165.0	-	0.1	yes	yes	166.0	-	0.1	yes	yes	165.0	18.8
40	3	4	5	0.1	yes	yes	182.0	-	0.0	yes	yes	198.0	-	0.1	yes	yes	181.0	17.1
40	3	8	1	0.0	yes	yes	157.0	-	0.0	yes	yes	173.0	-	0.1	yes	yes	156.0	11.5
40	3	8	2	0.0	yes	yes	180.0	-	0.0	yes	yes	181.0	-	0.1	yes	yes	180.0	12.2
40	3	8	3	0.1	yes	yes	197.0	-	0.0	yes	yes	164.0	-	0.2	yes	yes	197.0	7.6
40	3	8	4	0.0	yes	yes	168.0	-	0.0	yes	yes	163.0	-	0.1	yes	yes	167.0	18.6
40	3	8	5	0.0	yes	yes	198.0	-	0.0	yes	yes	199.0	-	0.1	yes	yes	198.0	14.1

Table B.18: Problem $V, Q|sd, r_i, d_i|\sum C_i$ (part 6)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
80	1	1	1	-	no	-	-	-	-	no	-	-	-	0.3	yes	yes	1821.0	7.5
80	1	1	2	0.1	yes	yes	1902.0	-	0.3	yes	yes	1951.0	-	0.2	yes	yes	1884.0	9.8
80	1	1	3	-	no	-	-	-	0.5	yes	yes	2013.0	-	0.3	yes	yes	2035.0	7.6
80	1	1	4	-	no	-	-	-	0.1	yes	yes	2063.0	-	0.2	yes	yes	1977.0	8.6
80	1	1	5	0.2	yes	yes	1861.0	-	0.2	yes	yes	1908.0	-	0.5	yes	yes	1838.0	4.5
80	1	2	1	0.2	yes	yes	1835.0	-	0.3	yes	yes	1757.0	-	0.5	yes	yes	1808.0	7.6
80	1	2	2	0.1	yes	yes	1762.0	-	0.8	yes	yes	1825.0	-	0.6	yes	yes	1737.0	4.5
80	1	2	3	0.3	yes	yes	1937.0	-	1.0	yes	yes	1797.0	-	0.5	yes	yes	1933.0	6.2
80	1	2	4	0.3	yes	yes	1764.0	-	0.4	yes	yes	1759.0	-	0.7	yes	yes	1756.0	7.0
80	1	2	5	0.2	yes	yes	1676.0	-	0.3	yes	yes	1690.0	-	0.5	yes	yes	1667.0	5.1
80	1	4	1	0.2	yes	yes	1784.0	-	0.4	yes	yes	1850.0	-	0.3	yes	yes	1779.0	4.6
80	1	4	2	0.5	yes	yes	1681.0	-	0.2	yes	yes	1699.0	-	0.7	yes	yes	1671.0	5.7
80	1	4	3	0.1	yes	yes	1732.0	-	0.1	yes	yes	1753.0	-	0.4	yes	yes	1726.0	4.4
80	1	4	4	0.8	yes	yes	1767.0	-	0.4	yes	yes	1795.0	-	2.0	yes	yes	1735.0	12.9
80	1	4	5	0.2	yes	yes	1788.0	-	0.1	yes	yes	1689.0	-	0.7	yes	yes	1779.0	6.7
80	1	8	1	0.1	yes	yes	1911.0	-	0.1	yes	yes	1695.0	-	0.3	yes	yes	1904.0	3.4
80	1	8	2	0.5	yes	yes	1694.0	-	0.1	yes	yes	1876.0	-	0.4	yes	yes	1687.0	4.2
80	1	8	3	0.2	yes	yes	1829.0	-	0.7	yes	yes	1653.0	-	0.3	yes	yes	1826.0	3.2
80	1	8	4	0.2	yes	yes	1567.0	-	0.7	yes	yes	1871.0	-	0.6	yes	yes	1566.0	6.6
80	1	8	5	0.1	yes	yes	1633.0	-	0.1	yes	yes	1771.0	-	0.3	yes	yes	1629.0	6.0
80	2	1	1	0.2	yes	yes	967.0	-	0.1	yes	yes	1034.0	-	0.5	yes	yes	954.0	6.8
80	2	1	2	0.2	yes	yes	955.0	-	0.1	yes	yes	981.0	-	1.7	yes	yes	918.0	9.9
80	2	1	3	-	no	-	-	-	0.1	yes	yes	1088.0	-	0.6	yes	yes	959.0	6.4
80	2	1	4	0.1	yes	yes	993.0	-	0.1	yes	yes	1117.0	-	0.4	yes	yes	977.0	9.9
80	2	1	5	0.1	yes	yes	991.0	-	0.1	yes	yes	974.0	-	0.7	yes	yes	976.0	14.4
80	2	2	1	0.4	yes	yes	909.0	-	1.1	yes	yes	900.0	5.0	0.6	yes	yes	901.0	10.8
80	2	2	2	0.2	yes	yes	894.0	-	0.5	yes	yes	820.0	-	1.2	yes	yes	887.0	11.5
80	2	2	3	0.7	yes	yes	944.0	-	0.7	yes	yes	948.0	-	3.1	yes	yes	939.0	11.2
80	2	2	4	4.8	yes	yes	920.0	-	0.4	yes	yes	921.0	-	3.8	yes	yes	910.0	7.5
80	2	2	5	0.1	yes	yes	1012.0	-	0.5	yes	yes	881.0	-	0.6	yes	yes	1012.0	6.8
80	2	4	1	0.1	yes	yes	947.0	-	0.2	yes	yes	856.0	-	0.5	yes	yes	946.0	5.7
80	2	4	2	0.1	yes	yes	851.0	-	0.2	yes	yes	836.0	-	0.5	yes	yes	850.0	7.1
80	2	4	3	0.2	yes	yes	901.0	4.3	1.3	yes	yes	888.0	-	0.4	yes	yes	896.0	6.3
80	2	4	4	0.4	yes	yes	900.0	-	0.2	yes	yes	915.0	-	0.9	yes	yes	896.0	11.0
80	2	4	5	0.7	yes	yes	944.0	-	0.2	yes	yes	909.0	-	2.1	yes	yes	942.0	5.8
80	2	8	1	0.3	yes	yes	894.0	-	0.1	yes	yes	937.0	-	0.7	yes	yes	892.0	5.6
80	2	8	2	0.3	yes	yes	973.0	-	0.6	yes	yes	847.0	-	0.8	yes	yes	971.0	6.7
80	2	8	3	0.3	yes	yes	870.0	-	0.1	yes	yes	937.0	-	0.6	yes	yes	870.0	6.1
80	2	8	4	0.2	yes	yes	862.0	-	0.1	yes	yes	907.0	-	0.6	yes	yes	862.0	6.3
80	2	8	5	0.2	yes	yes	957.0	-	0.1	yes	yes	878.0	-	0.8	yes	yes	956.0	7.9
80	3	1	1	-	no	-	-	-	0.1	yes	yes	717.0	-	0.7	yes	yes	643.0	14.3
80	3	1	2	0.1	yes	yes	679.0	-	0.1	yes	yes	655.0	-	0.4	yes	yes	677.0	12.0
80	3	1	3	-	no	-	-	-	0.3	yes	yes	707.0	-	0.6	yes	yes	720.0	11.4
80	3	1	4	-	no	-	-	-	0.1	yes	yes	742.0	-	0.7	yes	yes	667.0	11.2
80	3	1	5	0.1	yes	yes	641.0	-	0.1	yes	yes	682.0	-	0.6	yes	yes	636.0	11.6
80	3	2	1	1.4	yes	yes	610.0	-	0.8	yes	yes	663.0	-	6.4	yes	yes	609.0	8.4
80	3	2	2	0.2	yes	yes	704.0	-	0.7	yes	yes	632.0	-	0.8	yes	yes	703.0	8.5
80	3	2	3	3.1	yes	yes	625.0	-	0.1	yes	yes	606.0	-	2.2	yes	yes	622.0	10.9
80	3	2	4	0.4	yes	yes	588.0	-	0.3	yes	yes	618.0	-	1.3	yes	yes	588.0	9.9
80	3	2	5	0.1	yes	yes	645.0	-	0.3	yes	yes	652.0	-	0.5	yes	yes	644.0	9.8
80	3	4	1	0.1	yes	yes	665.0	-	0.1	yes	yes	599.0	-	0.6	yes	yes	665.0	7.1
80	3	4	2	0.1	yes	yes	626.0	-	0.1	yes	yes	595.0	-	0.6	yes	yes	626.0	10.1
80	3	4	3	0.4	yes	yes	638.0	-	0.1	yes	yes	628.0	-	1.0	yes	yes	638.0	6.9
80	3	4	4	0.8	yes	yes	675.0	-	0.7	yes	yes	652.0	-	3.9	yes	yes	673.0	9.8
80	3	4	5	0.1	yes	yes	653.0	-	0.6	yes	yes	756.0	-	0.6	yes	yes	652.0	9.4
80	3	8	1	0.2	yes	yes	622.0	-	0.3	yes	yes	674.0	-	0.9	yes	yes	620.0	6.6
80	3	8	2	0.1	yes	yes	558.0	-	0.1	yes	yes	659.0	-	0.4	yes	yes	558.0	10.2
80	3	8	3	0.1	yes	yes	663.0	-	0.3	yes	yes	648.0	-	0.7	yes	yes	663.0	6.6
80	3	8	4	0.4	yes	yes	616.0	-	0.1	yes	yes	627.0	-	1.1	yes	yes	616.0	13.8
80	3	8	5	0.3	yes	yes	709.0	-	0.2	yes	yes	619.0	-	0.8	yes	yes	708.0	6.8

Table B.19: Problem $V, Q|sd, r_i, d_i|\sum C_i$ (part 7)

n	V	Q	instance	w-t					w-w					w				
				CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)	CPU(s)	feas	opt	$\sum C_i$	gapGr(%)
160	1	1	1	2.5	yes	yes	7390.0	-	2.4	yes	yes	7784.0	-	7.6	yes	yes	7283.0	6.8
160	1	1	2	1.2	yes	yes	7697.0	-	1.8	yes	yes	7316.0	-	1.8	yes	yes	7636.0	4.5
160	1	1	3	2.0	yes	yes	6629.0	-	4.3	yes	yes	7319.0	-	3.8	yes	yes	6569.0	5.4
160	1	1	4	2.4	yes	yes	7457.0	-	2.3	yes	yes	7402.0	-	2.3	yes	yes	7408.0	8.1
160	1	1	5	0.7	yes	yes	7471.0	-	0.5	yes	yes	7348.0	-	1.4	yes	yes	7412.0	5.2
160	1	2	1	10.8	yes	yes	7361.0	-	34.9	yes	yes	6748.0	-	26.0	yes	yes	7351.0	4.9
160	1	2	2	0.6	yes	yes	6688.0	-	1.7	yes	yes	6900.0	-	2.1	yes	yes	6686.0	3.5
160	1	2	3	4.8	yes	yes	6776.0	-	1.3	yes	yes	6954.0	-	18.1	yes	yes	6767.0	3.7
160	1	2	4	11.2	yes	yes	6895.0	-	36.0	yes	yes	6117.0	-	14.8	yes	yes	6887.0	3.7
160	1	2	5	0.7	yes	yes	6439.0	-	1.2	yes	yes	6872.0	-	1.9	yes	yes	6436.0	3.8
160	1	4	1	1.4	yes	yes	6653.0	-	1.6	yes	yes	6369.0	-	1.8	yes	yes	6635.0	2.8
160	1	4	2	1.3	yes	yes	6856.0	-	6.5	yes	yes	6634.0	-	3.4	yes	yes	6847.0	3.7
160	1	4	3	3.7	yes	yes	6625.0	-	1.7	yes	yes	6881.0	-	4.3	yes	yes	6619.0	3.4
160	1	4	4	1.9	yes	yes	6799.0	-	6.6	yes	yes	6428.0	-	3.4	yes	yes	6795.0	3.8
160	1	4	5	0.6	yes	yes	6271.0	-	1.5	yes	yes	6374.0	-	1.4	yes	yes	6269.0	2.3
160	1	8	1	1.0	yes	yes	6832.0	-	2.3	yes	yes	6889.0	-	2.3	yes	yes	6826.0	2.9
160	1	8	2	0.8	yes	yes	6787.0	-	2.3	yes	yes	6492.0	-	2.4	yes	yes	6787.0	2.7
160	1	8	3	0.4	yes	yes	6134.0	-	0.7	yes	yes	6392.0	-	1.6	yes	yes	6132.0	1.8
160	1	8	4	1.2	yes	yes	6650.0	-	0.8	yes	yes	6686.0	-	2.2	yes	yes	6637.0	2.6
160	1	8	5	0.6	yes	yes	6300.0	-	0.9	yes	yes	6388.0	-	2.0	yes	yes	6293.0	3.3
160	2	1	1	5.3	yes	yes	3949.0	-	3.1	yes	yes	3773.0	-	37.7	yes	yes	3879.0	4.8
160	2	1	2	2.3	yes	yes	3716.0	-	2.0	yes	yes	3736.0	-	8.5	yes	yes	3700.0	6.3
160	2	1	3	0.6	yes	yes	3459.0	-	3.1	yes	yes	3729.0	-	3.0	yes	yes	3448.0	8.6
160	2	1	4	0.4	yes	yes	3917.0	-	2.2	yes	yes	3759.0	-	2.2	yes	yes	3909.0	5.7
160	2	1	5	2.7	yes	yes	3558.0	-	0.4	yes	yes	3783.0	-	9.9	yes	yes	3550.0	5.8
160	2	2	1	2.1	yes	yes	3590.0	-	20.6	yes	yes	3702.0	-	5.1	yes	yes	3585.0	4.4
160	2	2	2	2.6	yes	yes	3386.0	-	8.2	yes	yes	3432.0	-	6.4	yes	yes	3386.0	5.4
160	2	2	3	2.0	yes	yes	3521.0	-	5.1	yes	yes	3407.0	-	4.5	yes	yes	3519.0	3.9
160	2	2	4	4.6	yes	yes	3456.0	-	14.6	yes	yes	3366.0	-	4.7	yes	yes	3451.0	5.2
160	2	2	5	1.9	yes	yes	3474.0	-	1.0	yes	yes	3357.0	-	3.6	yes	yes	3473.0	4.3
160	2	4	1	9.8	yes	yes	3693.0	-	2.0	yes	yes	3648.0	-	17.8	yes	yes	3689.0	4.3
160	2	4	2	0.9	yes	yes	3692.0	-	7.5	yes	yes	3327.0	-	3.9	yes	yes	3691.0	3.9
160	2	4	3	0.6	yes	yes	3697.0	-	1.2	yes	yes	3519.0	-	4.4	yes	yes	3693.0	3.0
160	2	4	4	1.5	yes	yes	3322.0	-	1.0	yes	yes	3393.0	-	3.1	yes	yes	3318.0	3.0
160	2	4	5	1.1	yes	yes	3401.0	-	1.2	yes	yes	3408.0	-	3.8	yes	yes	3401.0	4.1
160	2	8	1	5.5	yes	yes	3492.0	-	6.7	yes	yes	3215.0	-	18.0	yes	yes	3492.0	4.7
160	2	8	2	0.5	yes	yes	3350.0	-	0.8	yes	yes	3487.0	-	2.4	yes	yes	3350.0	3.9
160	2	8	3	0.6	yes	yes	3231.0	-	0.5	yes	yes	3352.0	2.7	3.0	yes	yes	3227.0	3.3
160	2	8	4	1.4	yes	yes	3236.0	-	2.0	yes	yes	3257.0	-	3.1	yes	yes	3234.0	3.4
160	2	8	5	0.2	yes	yes	3378.0	-	0.5	yes	yes	3404.0	-	1.7	yes	yes	3378.0	3.0
160	3	1	1	5.0	yes	yes	2527.0	-	2.5	yes	yes	2482.0	-	42.0	yes	yes	2488.0	7.0
160	3	1	2	0.9	yes	yes	2479.0	-	2.0	yes	yes	2747.0	-	7.4	yes	yes	2464.0	9.6
160	3	1	3	1.0	yes	yes	2522.0	-	0.6	yes	yes	2465.0	-	3.3	yes	yes	2515.0	9.3
160	3	1	4	2.6	yes	yes	2374.0	-	1.1	yes	yes	2510.0	-	8.1	yes	yes	2365.0	5.8
160	3	1	5	0.4	yes	yes	2424.0	-	0.5	yes	yes	2376.0	-	2.9	yes	yes	2418.0	6.1
160	3	2	1	32.3	yes	yes	2384.0	-	25.6	yes	yes	2522.0	-	25.6	yes	yes	2381.0	7.6
160	3	2	2	46.7	yes	yes	2292.0	-	66.0	yes	yes	2468.0	-	79.4	yes	yes	2289.0	7.5
160	3	2	3	1.6	yes	yes	2338.0	-	2.5	yes	yes	2218.0	-	5.4	yes	yes	2335.0	6.9
160	3	2	4	0.8	yes	yes	2206.0	-	2.8	yes	yes	2417.0	-	4.5	yes	yes	2204.0	5.3
160	3	2	5	1.1	yes	yes	2307.0	-	0.9	yes	yes	2421.0	-	7.0	yes	yes	2306.0	5.3
160	3	4	1	4.3	yes	yes	2281.0	-	6.0	yes	yes	2230.0	-	5.8	yes	yes	2277.0	4.3
160	3	4	2	2.8	yes	yes	2342.0	-	1.7	yes	yes	2334.0	-	8.6	yes	yes	2342.0	5.6
160	3	4	3	1.5	yes	yes	2520.0	-	1.6	yes	yes	2356.0	-	5.8	yes	yes	2518.0	4.2
160	3	4	4	1.4	yes	yes	2216.0	-	0.7	yes	yes	2293.0	-	6.7	yes	yes	2216.0	6.0
160	3	4	5	0.9	yes	yes	2330.0	-	0.9	yes	yes	2268.0	-	5.3	yes	yes	2330.0	5.1
160	3	8	1	2.2	yes	yes	2301.0	-	0.9	yes	yes	2239.0	5.3	7.3	yes	yes	2300.0	6.3
160	3	8	2	1.7	yes	yes	2502.0	-	1.5	yes	yes	2307.0	-	10.4	yes	yes	2502.0	5.2
160	3	8	3	0.8	yes	yes	2255.0	-	0.8	yes	yes	2334.0	-	5.3	yes	yes	2255.0	4.1
160	3	8	4	1.0	yes	yes	2279.0	-	1.0	yes	yes	2385.0	-	5.0	yes	yes	2279.0	4.3
160	3	8	5	0.7	yes	yes	2148.0	-	0.7	yes	yes	2213.0	-	4.6	yes	yes	2148.0	4.7

Table B.20: Problem $V, Q|sd, r_i, d_i|\sum C_i$ (part 8)

Bibliography

Abedi, M., Chiong, R., Athauda, R., Seidgar, H., Michalewicz, Z., & Sturt, A. (2019). A regional multi-objective tabu search algorithm for a green heterogeneous dial-a-ride problem. *2019 IEEE Congress on Evolutionary Computation (CEC)*, 2082–2089. https://doi.org/10.1109/CEC.2019.8790003

Absi, N., Archetti, C., Dauzére-Pérés, S., & Feillet, D. (2015). A two-phase iterative heuristic approach for the production routing problem. *Transportation Science*, *49*(4), 784–795. https://doi.org/10.1287/trsc.2014.0523

Absi, N., Archetti, C., Dauzére-Pérés, S., Feillet, D., & Speranza, M. G. (2018). Comparing sequential and integrated approaches for the production routing problem. *European Journal of Operational Research*, *269*, 633–646. https://doi.org/10.1016/J.EJOR.2018.01.052

ACEA. (2022). Car and van CO2 targets: Charging infrastructure essential to meet member state ambition [[Online; accessed 15-July-2022]].

Adler, J. D., & Mirchandani, P. B. (2014). Online routing and battery reservations for electric vehicles with swappable batteries. *Transportation Research Part B: Methodological*, *70*, 285–302. https://doi.org/10.1016/J.TRB.2014.09.005

Adulyasak, Y., Cordeau, J. F., & Jans, R. (2015). The production routing problem: A review of formulations and solution algorithms. *Computers & Operations Research*, *55*, 141–152. https://doi.org/10.1016/J.COR.2014.01.011

Adulyasak, Y., Cordeau, J.-F., & Jans, R. (2014). Optimization-based adaptive large neighborhood search for the production routing problem. *Transportation Science*, *48*(1), 20–45. https://doi.org/10.1287/trsc.1120.0443

Ahmad, F., Saad Alam, M., Saad Alsaidan, I., & Shariff, S. M. (2020). Battery swapping station for electric vehicles: Opportunities and challenges. *IET Smart Grid*, *3*(3), 280–286. https://doi.org/https://doi.org/10.1049/iet-stg.2019.0059

Anenberg, S., Miller, J., Henze, D., & Minjares, R. (2019). A global snapshot of the air pollution-related health impacts of transportation sector emissions in 2010 and 2015. *International Council on Clean Transportation: Washington, DC, USA*.

Anily, S., & Pfeffer, A. (2013). The uncapacitated swapping problem on a line and on a circle. *Discrete Applied Mathematics*, *161*(4-5), 454–465.

Antonialli, F. (2019). International benchmark on experimentations with autonomous shuttles for collective transport. *27th International Colloquium of Gerpisa*.

Archetti, C., Christiansen, M., & Speranza, M. G. (2018). Inventory routing with pickups and deliveries. *European Journal of Operational Research*, *268*, 314–324. https://doi.org/10.1016/J.EJOR.2018.01.010

Archetti, C., Speranza, M. G., Boccia, M., Sforza, A., & Sterle, C. (2020). A branch-and-cut algorithm for the inventory routing problem with pickups and deliveries. *European Journal of Operational Research*, *282*, 886–895. https://doi.org/10.1016/J.EJOR.2019.09.056

Arora, S., & Barak, B. (2009). *Computational complexity: A modern approach*. Cambridge University Press. https://doi.org/10.1017/CBO9780511804090

Atahran, A., Lenté, C., & T'kindt, V. (2014). A multicriteria dial-a-ride problem with an ecological measure and heterogeneous vehicles. *Journal of Multi-Criteria Decision Analysis*, *21*(5-6), 279–298. https://doi.org/https://doi.org/10.1002/mcda.1518

Atallah, M. J., & Kosaraju, S. R. (1988). Efficient solutions to some transportation problems with applications to minimizing robot arm travel. *SIAM Journal on Computing*, *17*(5), 849–869.

Ausiello, G., Protasi, M., Marchetti-Spaccamela, A., Gambosi, G., Crescenzi, P., & Kann, V. (1999). *Complexity and approximation: Combinatorial optimization problems and their approximability properties* (1st). Springer-Verlag.

Baek, D., Chen, Y., Macii, E., Poncino, M., & Chang, N. (2019). Battery-aware electric truck delivery route planner. *2019 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, 1–6. https://doi.org/10.1109/ISLPED.2019.8824835

Bagloee, S. A., Tavana, M., Asadi, M., & Oliver, T. (2016). Autonomous vehicles: Challenges, opportunities, and future implications for transportation policies. *Journal of Modern Transportation*, *24*, 284–303. https://doi.org/10.1007/S40534-016-0117-3/FIGURES/5

Baïou, M., Colares, R., & Kerivin, H. (2018). The stop number minimization problem: Complexity and polyhedral analysis. *International Symposium on Combinatorial Optimization*, 64–76.

Barco, J., Guerra, A., Muñoz, L., & Quijano, N. (2017). Optimal routing and scheduling of charge for electric vehicles: A case study. *Mathematical Problems in Engineering*, *2017*, 1–16. https://EconPapers.repec.org/RePEc:hin:jnlmpe:8509783

Battarra, M., Cordeau, J.-F., & Iori, M. (2014). Chapter 6: Pickup-and-delivery problems for goods transportation. *Vehicle routing: Problems, methods, and applications, second edition* (pp. 161–191). SIAM.

Bendali, Fatiha, Mole Kamga, Eloise, Mailfert, Jean, Quilliot, Alain, & Toussaint, Helene. (2021). Synchronizing energy production and vehicle routing. *RAIRO-Oper. Res.*, *55*(4), 2141–2163. https://doi.org/10.1051/ro/2021093

Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I., & Laporte, G. (2007). Static pickup and delivery problems: A classification scheme and survey. *Top*, *15*(1), 1–31.

Berbeglia, G., Cordeau, J.-F., & Laporte, G. (2010). Dynamic pickup and delivery problems. *European journal of operational research*, *202*(1), 8–15.

Berge, C. (1985). *Graphs* [Rev. translation of: Graphes et hypergraphes, 1ère ptie: Graphes.]. North Holland.

Bongiovanni, C., Kaspi, M., & Geroliminis, N. (2019). The electric autonomous dial-a-ride problem. *Transportation Research Part B: Methodological*, *122*, 436–456. https://doi.org/10.1016/J.TRB.2019.03.004

Bonomo, F., Durán, G., & Marenco, J. (2009). Exploring the complexity boundary between coloring and list-coloring. *Annals of Operations Research*, *169*(1), 3.

Brůhová Foltýnová, H., Vejchodská, E., Rybová, K., & Květoň, V. (2020). Sustainable urban mobility: One definition, different stakeholders' opinions. *Transportation Research Part D: Transport and Environment*, *87*, 102465. https://doi.org/https://doi.org/10.1016/j.trd.2020.102465

Capros, P., De Vita, A., Tasios, N., Siskos, P., Kannavou, M., Petropoulos, A., Evangelopoulou, S., Zampara, M., Papadopoulos, D., Nakos, C., et al. (2016). Eu reference scenario 2016-energy, transport and ghg emissions trends to 2050.

Chabrol, M., Gourgand, M., & Leclaire, P. (2008). A coupling metaheuristic-simulation for a dial-a-ride problem with electric autonomy constraints in real traffic conditions. *Workshop on Metaheuristics for Logistics and Vehicle Routing, EU-IMEeting 2008*.

Chau, K. (2014). 21 - pure electric vehicles. In R. Folkson (Ed.), *Alternative fuels and advanced vehicle technologies for improved environmental performance* (pp. 655–684). Woodhead Publishing. https://doi.org/https://doi.org/10.1533/9780857097422.3.655

Chazelle, B. (2000). A minimum spanning tree algorithm with inverse-ackermann type complexity. *J. ACM*, *47*(6), 1028–1047. https://doi.org/10.1145/355541.355562

Chen, J., Qi, M., & Miao, L. (2016). The electric vehicle routing problem with time windows and battery swapping stations. *2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, 712–716. https://doi.org/10.1109/IEEM.2016.7797968

Chen, T., Zhang, B., Pourbabak, H., Kavousi-Fard, A., & Su, W. (2018). Optimal routing and charging of an electric vehicle fleet for high-efficiency dynamic transit systems. *IEEE Transactions on Smart Grid*, *9*(4), 3563–3572. https://doi.org/10.1109/TSG.2016.2635025

Cheng, Y., & Tao, J. (2018). Optimization of a micro energy network integrated with electric bus battery swapping station and distributed pv. *2018 2nd IEEE Conference on Energy Internet and Energy System Integration (EI2)*, 1–6. https://doi.org/10.1109/EI2.2018.8582236

Christodoulou, A., & Christidis, P. (2020). Measuring congestion in european cities. *Publications Office of the European Union, Luxembourg, URL: https://publications. jrc. ec. europa. eu/repository/handle/JRC118448*.

Coelho, L. C., Cordeau, J.-F., & Laporte, G. (2014). Thirty years of inventory routing. *Transportation Science*, *48*(1), 1–19. https://doi.org/10.1287/trsc.2013.0472

Commission, E. (2011). *Roadmap to a single european transport area: Towards a competitive and resource efficient transport system: White paper*. Publications Office of the European Union.

Cordeau, J.-F., & Laporte, G. (2007). The dial-a-ride problem: Models and algorithms. *Annals of operations research*, *153*(1), 29–46.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to algorithms, 3rd edition*. MIT Press. http://mitpress.mit.edu/books/introduction-algorithms

Desaulniers, G., Desrosiers, J., Erdmann, A., Solomon, M. M., & Soumis, F. (2002). 9. vrp with pickup and delivery. *The vehicle routing problem* (pp. 225–242). https://doi.org/10.1137/1.9780898718515.ch9

Ding, R., Liu, Z., Li, X., Hou, Y., Sun, W., Zhai, H., & Wei, X. (2022). Joint charging scheduling of electric vehicles with battery to grid technology in battery swapping station [2021 International Conference on New Energy and Power Engineering]. *Energy Reports*, *8*, 872–882. https://doi.org/https://doi.org/10.1016/j.egyr.2022.02.029

Doerner, K. F., & Salazar-González, J.-J. (2014). Chapter 7: Pickup-and-delivery problems for people transportation. *Vehicle routing: Problems, methods, and applications, second edition* (pp. 193–212). SIAM.

Dror, M., Ball, M., & Golden, B. (1985). A computational comparison of algorithms for the inventory routing problem. *Annals of Operations Research*, *4*, 1–23. https://doi.org/10.1007/BF02022035

EEA. (2013). Annual european union greenhouse gas inventory 1990–2011 and inventory report 2013. submission to the unfccc secretariat.

ENVI. (2022). Fit for 55: MEPs back objective of zero emissions for cars and vans in 2035 [[Online; accessed 15-July-2022]].

Erdelić, T., & Carić, T. (2019). A survey on the electric vehicle routing problem: Variants and solution approaches. *Journal of Advanced Transportation*, *2019*, 5075671. https://doi.org/10.1155/2019/5075671

Even, S., Itai, A., & Shamir, A. (1976). On the complexity of timetable and multicommodity flow problems. *SIAM Journal on Computing*, *5*. https://doi.org/10.1137/0205048

Fiori, C., & Marzano, V. (2018). Modelling energy consumption of electric freight vehicles in urban pickup/delivery operations: Analysis and estimation on a real-world dataset. *Transportation Research Part D: Transport and Environment*, *65*, 658–673. https://doi.org/10.1016/J.TRD.2018.09.020

Frederickson, G. N. (1993). A note on the complexity of a simple transportation problem. *SIAM Journal on Computing*, *22*(1), 57–61.

Garey, M. R., Johnson, D. S., Miller, G. L., & Papadimitriou, C. H. (1980). The complexity of coloring circular arcs and chords. *SIAM Journal on Algebraic Discrete Methods*, *1*. https://doi.org/10.1137/0601025

Gendreau, M., Laporte, G., & Vigo, D. (1999). Heuristics for the traveling salesman problem with pickup and delivery. *Computers & Operations Research*, *26*(7), 699–714.

Ghobadi, A., Tavakkoli Moghadam, R., Fallah, M., & Kazemipoor, H. (2021). Multi-depot electric vehicle routing problem with fuzzy time windows and pick-up/delivery constraints. *Journal of Applied Research on Industrial Engineering*, *8*(1), 1–18. https://doi.org/10.22105/jarie.2021.231764.1165

Goeke, D. (2019). Granular tabu search for the pickup and delivery problem with time windows and electric vehicles. *European Journal of Operational Research*, *278*, 821–836. https://doi.org/10.1016/J.EJOR.2019.05.010

Gogoro. (2017). Solar Power is Charging Batteries [[Online; accessed 15-July-2022]].

Gökay, S., Heuvels, A., & Krempels, K.-H. (2019). A high-level category survey of dial-a-ride problems. *VEHITS*, 594–600.

Golsefidi, A. H., & Jokar, M. R. A. (2020). A robust optimization approach for the production-inventory-routing problem with simultaneous pickup and delivery. *Computers & Industrial Engineering*, *143*, 106388. https://doi.org/10.1016/J.CIE.2020.106388

Gonçalves, F., Cardoso, S. R., Relvas, S., & Barbosa-Póvoa, A. (2011). Optimization of a distribution network using electric vehicles: A vrp problem. *Proceedings of the IO2011-15 Congresso da associação Portuguesa de Investigação Operacional, Coimbra, Portugal*, 18–20.

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Kan, A. R. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of discrete mathematics* (pp. 287–326). Elsevier.

Grandinetti, L., Guerriero, F., Pezzella, F., & Pisacane, O. (2016). A pick-up and delivery problem with time windows by electric vehicles. *International Journal of Productivity and Quality Management*, *18*, 403–423. https://EconPapers.repec.org/RePEc:ids:ijpqma:v:18:y:2016:i:2/3:p:403-423

Guan, D. J. (1998). Routing a vehicle of capacity greater than one. *Discrete Applied Mathematics*, *81*(1-3), 41–57.

Hancock, P. A., Nourbakhsh, I., & Stewart, J. (2019). On the future of transportation in an era of automated and autonomous vehicles. *Proceedings of the National Academy of Sciences*, *116*(16), 7684–7691. https://doi.org/10.1073/pnas.1805770115

Helmers, E., & Marx, P. (2012). Electric cars: Technical characteristics and environmental impacts. *Environmental Sciences Europe*, *24*, 14. https://doi.org/10.1186/2190-4715-24-14

Herrmann, F., & Rothfuss, F. (2015). Introduction to hybrid electric vehicles, battery electric vehicles, and off-road electric vehicles. *Advances in Battery Technologies for Electric Vehicles*, 3–16. https://doi.org/10.1016/B978-1-78242-377-5.00001-7

Ho, S. C., Szeto, W., Kuo, Y.-H., Leung, J. M., Petering, M., & Tou, T. W. (2018). A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological*, *111*, 395–421.

Hof, J., Schneider, M., & Goeke, D. (2017). Solving the battery swap station location-routing problem with capacitated electric vehicles using an avns algorithm for vehicle-routing problems with intermediate stops. *Transportation Research Part B: Methodological*, *97*, 102–112. https://doi.org/10.1016/J.TRB.2016.11.009

Hosseini, Z. S., Mahoor, M., & Khodaei, A. (2018). Battery swapping station as an energy storage for capturing distribution-integrated solar variability. *2018 North American Power Symposium (NAPS)*, 1–6. https://doi.org/10.1109/NAPS.2018.8600598

Iassinovskaia, G., Limbourg, S., & Riane, F. (2017). The inventory-routing problem of returnable transport items with time windows and simultaneous pickup and delivery in closed-loop supply chains. *International Journal of Production Economics*, *183*, 570–582. https://doi.org/10.1016/J.IJPE.2016.06.024

Ilani, H., Shufan, E., & Grinshpoun, T. (2015). A fixed route dial-a-ride problem. *Proceedings of the 7th Multidisciplinary International Conference on Scheduling : Theory and Applications (MISTA 2015), 25 - 28 Aug 2015, Prague, Czech Republic*, 313–324.

Itai, A., Perl, Y., & Shiloach, Y. (1982). The complexity of finding maximum disjoint paths with length constraints. *Networks*, *12*(3), 277–286.

Jie, W., Yang, J., Zhang, M., & Huang, Y. (2019). The two-echelon capacitated electric vehicle routing problem with battery swapping stations: Formulation and efficient methodology. *European Journal of Operational Research*, *272*, 879–904. https://doi.org/10.1016/J.EJOR.2018.07.002

Jung, J., & Jayakrishnan, R. (2012). High-coverage point-to-point transit: Electric vehicle operations. *Transportation Research Record*, *2287*(1), 44–53. https://doi.org/10.3141/2287-06

Jungnickel, D. (2013). Algorithms and complexity. *Graphs, networks and algorithms* (pp. 35–63). Springer.

Justin, C. Y., Payan, A. P., Briceno, S. I., German, B. J., & Mavris, D. N. (2020). Power optimized battery swap and recharge strategies for electric aircraft operations. *Transportation Research Part C: Emerging Technologies*, *115*, 102605. https://doi.org/https://doi.org/10.1016/j.trc.2020.02.027

Kang, Q., Wang, J., Zhou, M., & Ammari, A. C. (2016). Centralized charging strategy and scheduling algorithm for electric vehicles under a battery swapping sce-

nario. *IEEE Transactions on Intelligent Transportation Systems*, *17*(3), 659–669. https://doi.org/10.1109/TITS.2015.2487323

Kaspi, M., Raviv, T., & Ulmer, M. W. (2022). Directions for future research on urban mobility and city logistics. *Networks*, *79*(3), 253–263. https://doi.org/https://doi.org/10.1002/net.22092

Keolis deploys electric autonomous shuttles at two university campuses in France [[Online; accessed 04-October-2019]]. (2018).

Kleinberg, J., & Tardos, E. (2006). *Algorithm design*. Pearson Education India.

Koç, Ç., Jabali, O., Mendoza, J. E., & Laporte, G. (2019). The electric vehicle routing problem with shared charging stations. *International Transactions in Operational Research*, *26*(4), 1211–1243. https://doi.org/https://doi.org/10.1111/itor.12620

Li, H., & Lim, A. (2003). A metaheuristic for the pickup and delivery problem with time windows. *International Journal on Artificial Intelligence Tools*, *12*(02), 173–186. https://doi.org/10.1142/S0218213003001186

Li, J., Wang, F., & He, Y. (2020). Electric vehicle routing problem with battery swapping considering energy consumption and carbon emissions. *Sustainability*, *12*(24). https://doi.org/10.3390/su122410537

Lin, J., Zhou, W., & Wolfson, O. (2016). Electric vehicle routing problem. *Transportation Research Procedia*, *12*, 508–521. https://doi.org/10.1016/J.TRPRO.2016.02.007

Litman, T. (2017). *Autonomous vehicle implementation predictions*. Victoria Transport Policy Institute Victoria, BC, Canada.

Liu, N., Chen, Q., Lu, X., Liu, J., & Zhang, J. (2015). A charging strategy for pv-based battery switch stations considering service availability and self-consumption of pv energy. *IEEE Transactions on Industrial Electronics*, *62*(8), 4878–4889. https://doi.org/10.1109/TIE.2015.2404316

Lu, C. C., Yan, S., & Huang, Y. W. (2018). Optimal scheduling of a taxi fleet with mixed electric and gasoline vehicles to service advance reservations. *Transportation Research Part C: Emerging Technologies*, *93*, 479–500. https://doi.org/10.1016/J.TRC.2018.06.015

Ma, T.-Y. (2021). Two-stage battery recharge scheduling and vehicle-charger assignment policy for dynamic electric dial-a-ride services. *PLOS ONE*, *16*(5), 1–27. https://doi.org/10.1371/journal.pone.0251582

Maggi, L., & Faizrahnemoon, M. (2014). Demo: Dial-a-ride: A green shortest path algorithm. *Proceedings of the 15th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 413–414. https://doi.org/10.1145/2632951.2636058

Mahoor, M., Hosseini, Z. S., & Khodaei, A. (2019). Least-cost operation of a battery swapping station with random customer requests. *Energy*, *172*, 913–921. https://doi.org/https://doi.org/10.1016/j.energy.2019.02.018

Manyika, J., Chui, M., Bughin, J., Dobbs, R., Bisson, P., & Marrs, A. (2013). *Disruptive technologies: Advances that will transform life, business, and the global economy* (Vol. 180). McKinsey Global Institute San Francisco, CA.

Mao, H., Shi, J., Zhou, Y., & Zhang, G. (2020). The electric vehicle routing problem with time windows and multiple recharging options. *IEEE Access*, *8*, 114864–114875. https://doi.org/10.1109/ACCESS.2020.3003000

Marx, D. (2003). *A short proof of the np-completeness of circular arc coloring.*

Marx, D. (2005). A short proof of the np-completeness of minimum sum interval coloring. *Operations Research Letters*, *33*(4), 382–384.

Masmoudi, M. A., Hosny, M., & Demir, E. (2019). An adaptive large neighborhood search heuristic for the green dial-a-ride problem. *Solving transport problems* (pp. 1–26). John Wiley & Sons, Ltd. https://doi.org/https://doi.org/10.1002/9781119686750.ch1

Masmoudi, M. A., Hosny, M., Demir, E., Genikomsakis, K. N., & Cheikhrouhou, N. (2018). The dial-a-ride problem with electric vehicles and battery swapping stations. *Transportation Research Part E: Logistics and Transportation Review*, *118*, 392–420. https://doi.org/https://doi.org/10.1016/j.tre.2018.08.005

Masmoudi, M. A., Hosny, M., Demir, E., & Pesch, E. (2020). Hybrid adaptive large neighborhood search algorithm for the mixed fleet heterogeneous dial-a-ride problem. *Journal of Heuristics*, *26*, 83–118. https://doi.org/10.1007/s10732-019-09424-x

McKinsey. (2022). Why the automotive future is electric [[Online; accessed 15-July-2022]].

Miskolczi, M., Földes, D., Munkácsy, A., & Jászberényi, M. (2021). Urban mobility scenarios until the 2030s. *Sustainable Cities and Society*, *72*, 103029. https://doi.org/https://doi.org/10.1016/j.scs.2021.103029

Molenbruch, Y., Braekers, K., & Caris, A. (2017). Typology and literature review for dial-a-ride problems. *Annals of Operations Research*, *259*(1-2), 295–325.

Murty, U., & Bondy, A. (2008). Graph theory (graduate texts in mathematics 244).

Ni, L., Sun, B., Tan, X., & Tsang, D. H. K. (2021). Inventory planning and real-time routing for network of electric vehicle battery-swapping stations. *IEEE Transactions on Transportation Electrification*, *7*(2), 542–553. https://doi.org/10.1109/TTE.2020.3015290

Nurre, S. G., Bent, R., Pan, F., & Sharkey, T. C. (2014). Managing operations of plug-in hybrid electric vehicle (phev) exchange stations for use with a smart grid. *Energy Policy*, *67*, 364–377. https://doi.org/https://doi.org/10.1016/j.enpol.2013.11.052

Outalha, Abdelhak, Lakhal, Yassine, Baghli, Fatima Zahra, & Kzaiber, Fouzia. (2021). The efficiency of discrete event systems for the general pickup and delivery problem with electric vehicles. *E3S Web Conf.*, *229*, 01011. https://doi.org/10.1051/e3sconf/202122901011

Paris-Saclay Autonomous Lab: new autonomous, electric and shared mobility services [[Online; accessed 04-October-2019]]. (2019).

Parragh, S. N., Doerner, K. F., & Hartl, R. F. (2008a). A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, *58*, 21–51. https://doi.org/10.1007/s11301-008-0033-7

Parragh, S. N., Doerner, K. F., & Hartl, R. F. (2008b). A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, *58*, 81–117. https://doi.org/10.1007/s11301-008-0036-4

Pelletier, S., Jabali, O., & Laporte, G. (2016). 50th anniversary invited article-goods distribution with electric vehicles: Review and research perspectives. *Transportation Science*, *50*(1), 3–22. https://doi.org/10.1287/trsc.2015.0646

Perera, T., Prakash, A., Gamage, C. N., & Srikanthan, T. (2018). Hybrid genetic algorithm for an on-demand first mile transit system using electric vehicles. In Y. Shi, H. Fu, Y. Tian, V. V. Krzhizhanovskaya, M. H. Lees, J. Dongarra, & P. M. A. Sloot (Eds.), *Computational science – iccs 2018* (pp. 98–113). Springer International Publishing.

Pimenta, V., Quilliot, A., Toussaint, H., & Vigo, D. (2017a). Models and algorithms for reliability-oriented dial-a-ride with autonomous electric vehicles. *European Journal of Operational Research*, *257*(2), 601–613.

Pimenta, V., Quilliot, A., Toussaint, H., & Vigo, D. (2017b). Models and algorithms for reliability-oriented dial-a-ride with autonomous electric vehicles. *European Journal of Operational Research*, *257*, 601–613. https://doi.org/10.1016/J.EJOR.2016.07.037

Projet AVENUE : Navettes autonomes en milieu urbain [[Online; accessed 22-January-2021]]. (2020).

PVGIS. (2022). Photovoltaic Geographical Information System [[Online; accessed 15-July-2022]].

Qin, H., Su, X., Ren, T., & Luo, Z. (2021). A review on the electric vehicle routing problems: Variants and algorithms. *Frontiers of Engineering Management*, *8*, 370–389. https://doi.org/10.1007/s42524-021-0157-1

Raeesi, R., & Zografos, K. G. (2020). The electric vehicle routing problem with time windows and synchronised mobile battery swapping. *Transportation Research Part B: Methodological*, *140*, 101–129. https://doi.org/10.1016/J.TRB.2020.06.012

Revankar, S. R., & Kalkhambkar, V. N. (2021). Grid integration of battery swapping station: A review. *Journal of Energy Storage*, *41*, 102937. https://doi.org/https://doi.org/10.1016/j.est.2021.102937

Rüther, C., & Rieck, J. (2020). A grouping genetic algorithm for multi depot pickup and delivery problems with time windows and heterogeneous vehicle fleets. *European Conference on Evolutionary Computation in Combinatorial Optimization (Part of EvoStar)*, 148–163.

Sayarshad, H. R., Mahmoodian, V., & Gao, H. O. (2020). Non-myopic dynamic routing of electric taxis with battery swapping stations. *Sustainable Cities and Society*, *57*, 102113. https://doi.org/10.1016/J.SCS.2020.102113

Soysal, M., Çimen, M., & Belbağ, S. (2020). Pickup and delivery with electric vehicles under stochastic battery depletion. *Computers & Industrial Engineering*, *146*, 106512. https://doi.org/10.1016/J.CIE.2020.106512

Szkaliczki, T. (1999). Routing with minimum wire length in the dogleg-free manhattan model is $\mathcal{NP}$-complete. *SIAM Journal on Computing*, *29*(1), 274–287.

Tomaszewska, A., Chu, Z., Feng, X., O'Kane, S., Liu, X., Chen, J., Ji, C., Endler, E., Li, R., Liu, L., Li, Y., Zheng, S., Vetterlein, S., Gao, M., Du, J., Parkes, M., Ouyang, M., Marinescu, M., Offer, G., & Wu, B. (2019). Lithium-ion battery fast charging: A review. *eTransportation*, *1*, 100011. https://doi.org/https://doi.org/10.1016/j.etran.2019.100011

Toth, P., Vigo, D., Toth, P., & Vigo, D. (2014). *Vehicle routing: Problems, methods, and applications, second edition*. Society for Industrial; Applied Mathematics.

Transport in the European Union: Current Trends and Issues [[Online; accessed 10-June-2021]]. (2019).

Trotta, M., Archetti, C., Feillet, D., & Quilliot, A. (2021). *The cumulative cost pickup and delivery problem on rings* (Working Paper No. 2021/10). MSE CMP-SFL.

Trotta, M., Archetti, C., Feillet, D., & Quilliot, A. (2022). Pickup and delivery problems with autonomous vehicles on rings. *European Journal of Operational Research*, *300*(1), 221–236. https://doi.org/https://doi.org/10.1016/j.ejor.2021.07.050

Tzoreff, T. E., Granot, D., Granot, F., & Sošić, G. (2002). The vehicle routing problem with pickups and deliveries on some special graphs. *Discrete Applied Mathematics*, *116*(3), 193–229.

Ulrich, L. (2021). How Is This A Good Idea?: EV Battery Swapping [[Online; accessed 15-July-2021]].

van Anholt, R. G., Coelho, L. C., Laporte, G., & Vis, I. F. A. (2016). An inventory-routing problem with pickups and deliveries arising in the replenishment of automated teller machines. *Transportation Science*, *50*(3), 1077–1091. https://doi.org/10.1287/trsc.2015.0637

van Essen, H., & Kampman, B. (2011). Impacts of electric vehicles – summary report.

Verma, A. (2018). Electric vehicle routing problem with time windows, recharging stations and battery swapping stations. *EURO Journal on Transportation and Logistics*, *7*, 415–451. https://doi.org/10.1007/S13676-018-0136-9

Wahlström, M. (2018). Euler digraphs. *Classes of directed graphs* (pp. 173–205). Springer.

Wang, H., & Cheu, R. L. (2013). Operations of a taxi fleet for advance reservations using electric vehicles and charging stations. *Transportation Research Record*, *2352*(1), 1–10. https://doi.org/10.3141/2352-01

Widrick, R. S., Nurre, S. G., & Robbins, M. J. (2018). Optimal policies for the management of an electric vehicle battery swap station. *Transportation Science*, *52*(1), 59–79. https://doi.org/10.1287/trsc.2016.0676

Worley, O., Klabjan, D., & Sweda, T. M. (2012). Simultaneous vehicle routing and charging station siting for commercial electric vehicles. *2012 IEEE International Electric Vehicle Conference*, 1–3. https://doi.org/10.1109/IEVC.2012.6183279

Wu, J., Zheng, L., Huang, C., Cai, S., Feng, S., & Zhang, D. (2019). An improved hybrid heuristic algorithm for pickup and delivery problem with three-dimensional loading constraints. *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, 1607–1612.

Xiao, Y., Zuo, X., Kaku, I., Zhou, S., & Pan, X. (2019). Development of energy consumption optimization model for the electric vehicle routing problem with time windows. *Journal of Cleaner Production*, *225*, 647–663. https://doi.org/10.1016/J.JCLEPRO.2019.03.323

Yang, H., Yang, S., Xu, Y., Cao, E., Lai, M., & Dong, Z. (2015). Electric vehicle route optimization considering time-of-use electricity price by learnable partheno-genetic algorithm. *IEEE Transactions on Smart Grid*, *6*(2), 657–666. https://doi.org/10.1109/TSG.2014.2382684

Yang, J., & Sun, H. (2015). Battery swap station location-routing problem with capacitated electric vehicles. *Computers & Operations Research*, *55*, 217–232. https://doi.org/https://doi.org/10.1016/j.cor.2014.07.003

Yang, Q.-q., Hu, D.-w., Chu, H.-f., & Xu, C.-r. (2018). An electric vehicle routing problem with pickup and delivery. *Cictp 2018: Intelligence, connectivity, and mobility* (pp. 176–184). American Society of Civil Engineers Reston, VA.

Yu, Y., Wu, Y., & Wang, J. (2019). Bi-objective green ride-sharing problem: Model and exact method. *International Journal of Production Economics*, *208*, 472–482. https://doi.org/10.1016/J.IJPE.2018.12.007