

O PROBLEMA DA ÁRVORE DE CAMINHOS
MAIS CURTOS ROBUSTA: FORMULAÇÕES E
ALGORITMOS

IAGO AUGUSTO DE CARVALHO

O PROBLEMA DA ÁRVORE DE CAMINHOS
MAIS CURTOS ROBUSTA: FORMULAÇÕES E
ALGORITMOS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

ORIENTADOR: THIAGO FERREIRA DE NORONHA
COORIENTADOR: LUIZ FILIPE MENEZES VIEIRA

Belo Horizonte
Fevereiro de 2016

IAGO AUGUSTO DE CARVALHO

**THE ROBUST SHORTEST PATH TREE
PROBLEM: FORMULATIONS AND
ALGORITHMS**

Dissertation presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

ADVISOR: THIAGO FERREIRA DE NORONHA
CO-ADVISOR: LUIZ FILIPE MENEZES VIEIRA

Belo Horizonte

February 2016

Ficha catalográfica elaborada pela Biblioteca do ICEx - UFMG

Carvalho, Iago Augusto de.

C331r The robust shortest path tree problem: formulations and algorithms. / Iago Augusto de Carvalho. – Belo Horizonte, 2016.

xvi, 42 f.: il.; 29 cm.

Dissertação (mestrado) - Universidade Federal de Minas Gerais – Departamento de Ciência da Computação.

Orientador: Thiago Ferreira de Noronha.

Coorientador: Luiz Felipe Menezes Vieira.

1. Computação - Teses. 2. Otimização matemática
3. Otimização robusta. 4. Internet das Coisas. I. Orientador.
II. Coorientador. III. Título.

CDU 519.6*61(043)



UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO


FOLHA DE APROVAÇÃO

The robust shortest path tree problem: formulations and algorithms

IAGO AUGUSTO DE CARVALHO

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:


PROF. THIAGO FERREIRA DE NORONHA - Orientador
Departamento de Ciência da Computação - UFMG


PROF. LUIZ FILIPE MENEZES VIEIRA - Coorientador
Departamento de Ciência da Computação - UFMG

PROF. CHRISTOPHE DIDIER DUHAMEL
Université Blaise Pascal


PROF. GERALDO ROBSON MATEUS
Departamento de Ciência da Computação - UFMG



Belo Horizonte, 15 de fevereiro de 2016.

Resumo

IPv6 Low Wireless Personal Area Networks (6LoWPAN) é a mais promissora tecnologia para a implementação da chamada Internet das Coisas. Para que esta tecnologia torne-se uma realidade, protocolos de roteamento precisam ser resilientes a variações na qualidade da transmissão, devido a constantes mudanças nos enlaces. O mais promissor destes protocolos é o *IPv6 Routing Protocol for Low-Power and Lossy Networks* (RPL). Nesta dissertação, o protocolo RPL é estendido de forma a considerar a incerteza na qualidade dos enlaces. O problema de roteamento do RPL Robusto é modelado como um problema de otimização robusta derivado do Problema da Árvore de Caminhos Mais Curtos, denominado Árvore de Caminhos Mais Curtos Robusta (RSPT). É propostas uma nova heurísticas para o RSPT, além de uma formulação matemática e um algoritmo exato baseado na formulação proposta. Além disso, uma heurística e três algoritmos aproximativos da literatura para problemas de Otimização Robusta são estendidos para o RSPT, e uma prova de seus fatores de aproximação foi desenvolvida. O algoritmo proposto é comparado com os algoritmos da literatura. Experimentos computacionais demonstram que o algoritmo exato proposto resolveu todas as instâncias propostas com 100 vértices na otimalidade. Entretanto, ele não consegue resolver instâncias com 200 vértices na otimalidade em um tempo de 24 horas. A heurística proposta apresenta melhores resultados que os algoritmos aproximativos estendidos da literatura, sendo que obtém um gap relativo próximo ao gap do algoritmo exato proposto com um tempo computacional muito inferior. A heurística proposta pode ser facilmente estendida para outros problemas de otimização robusta.

Palavras-chave: Otimização robusta, programação matemática, heurísticas, Internet das Coisas, RPL.

Abstract

IPv6 Low Wireless Personal Area Networks (6LoWPAN) is the most promising technology for implementing the so called Internet of Things. In order for this technology to become a reality, routing protocols need to be resilient to variations in the links quality, due the constantly changes in the channels. The most promising of these protocols is the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL). In this work, the RPL routing protocol was extended to consider the uncertainty in the link quality. The RPL routing problem is modeled as a Robust Optimization problem derived from the Shortest Path Tree problem, denominated Robust Shortest Path Tree problem (RSPT). A new heuristics for the RSPT is developed, besides a mathematical formulation and an exact algorithm based in the proposed formulation. Besides that, a heuristic and three aproximative algorithms from literature of Robust Optimization were extend for the RSPT, and a proof of its approximation ratio is developed. The proposed algorithms are compared with the algorithms from the literature. Computational experiments shown that the proposed exact algorithm solved all the proposed instances with 100 vertices at optimality. However, it could not solve instances with 200 vertices at optimality within 24 hours. The proposed heuristics presented better results that the approximative algorithms extended from literature, such that it achieves a relatively gap close to the gap of the proposed exact algorithm with a smaller computational time. The proposed heuristic can be easily extended to other robust optimization problems.

Keywords: Robust optimization, mathematical programming, heuristics, Internet of Things, RPL.

List of Figures

1.1	(a) An example of a DODAG rooted on node a , and (b) an example of a routing tree in this DODAG. In this example, a package from node e to node f follows the path $\langle e, c, a, d, f \rangle$	2
4.1	Examples of a RSPT instance (a) and one possible scenario (b)	13
6.1	A Karaşan graph with $M = m$ layers and width $W = 2$ [Karaşan et al., 2001]	26

List of Tables

2.1	Smart environment application domains [Gubbi et al., 2013]	5
6.1	Results for CPLEX Branch-and-bound Karařan instances with 100 nodes .	27
6.2	Results for CPLEX Branch-and-bound for Karařan instances with 200 nodes	27
6.3	Results for AM, AU and AMU for Karařan instances with 100 nodes, compared with the CPLEX branch-and-bound	28
6.4	Results for AM, AU and AMU for Karařan instances with 200 nodes, compared with the CPLEX branch-and-bound	28
6.5	Results for different versions of SBA	29
6.6	Results for CPLEX branch-and-bound, AMU and two different SBAs for Karařan instances with 200 vertices	30
6.7	Results for CPLEX branch-and-bound and MILP-VND for Karařan instances with 200 vertices	31
6.8	Results for CPLEX Branch-and-bound and heuristics for Karařan instances with 200 nodes	32

Contents

Resumo	ix
Abstract	xi
List of Figures	xiii
List of Tables	xv
1 Introduction	1
2 Routing in 6LoWPAN	5
3 Related works	9
4 The Robust Shortest Path Tree	13
4.1 The problem	13
4.1.1 RSPT complexity proof	14
4.2 A mixed integer linear programming formulation	17
5 Heuristics for RSPT	19
5.1 Proof of AM 2-approximation for RSPT	20
5.2 MILP-based Variable Neighbourhood Descent	22
6 Computational Experiments	25
6.1 CPLEX branch-and-bound	26
6.2 Average Median Upper algorithm	26
6.3 Scenario-based Algorithm	29
6.4 Mixed Integer Linear Programming Variable Neighbourhood Descent .	31
6.5 Comparison of heuristics	32
7 Conclusions	33

Chapter 1

Introduction

Nowadays, there is a steady growth in the number of Internet-connected devices such as computers, sensors, actuators, smartphones, home appliances, etc. [Atzori et al., 2010]. This new set of devices introduces a novel paradigm in the scenario of modern wireless telecommunications. These devices communicate with other and collaborate with their neighbours to reach common goals, forming the Internet of Things (IoT) [Giusto et al., 2010].

IoT refers to the networked interconnection of daily use objects, thus leading to a ubiquitous system. This ubiquity of the Internet is obtained by integrating objects via embedded systems. This leads to a highly distributed network of devices that communicate with each other and with people at their surrounding [Xia et al., 2012]. It can be characterized as a highly dynamic and distributed network system, composed of a large number of smart objects that produce and consume information [Miorandi et al., 2012].

A large number of applications are being developed for the IoT, with promises that this new paradigm will improve our quality of life [Xia et al., 2012]. However, many challenges, such as energy consumption, security and well designed routing protocols need to be solved before IoT becomes a reality. One of the most promising technologies for the implementation of IoT is the IPv6 Low Wireless Personal Area Network (6LoWPAN) [Shelby and Bormann, 2011]. It is characterized by low resources in terms of both computation and energy capacity [Atzori et al., 2010; Gubbi et al., 2013]. Each 6LoWPAN node represents a device in the IoT. These nodes are interconnected by wireless links with potentially low communication quality and high loss rates [Winter, 2012].

Many routing protocols for 6LoWPAN were designed in an attempt to overcome the aforementioned deficiencies. Currently, the most promising of these protocols is

the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [Winter, 2012; Gaddour and Koubâa, 2013]. For each application running at the network, a sink node s is specified. Then, RPL builds a Destination Oriented Direct Acyclic Graph (DODAG) from s to all other nodes serving the application. The DODAG represents all the possible routes from the sink to any other node in the application, and vice-versa. It is built taking into account the nodes' transmission range and the distance between nodes. Each node may have one or more parent nodes, and the arcs are oriented from the node to its parents.

Once the DODAG is built, a default parent is selected for each node, which induce a s -rooted tree that is a subgraph of the DODAG. This tree specifies the default routes between nodes in a 6LoWPAN, and its is referred to as the *routing tree*. It is computed taking into account a predefined metric regarding the link quality. There are an exponentially large number of s -rooted trees in a DODAG, and the efficiency of the network depends on how good is the one chosen by the protocol. Therefore, each node periodically updates its parent, inducing a possibly better routing tree. An example of a DODAG and a routing tree is presented in Figure 1.1. In this example, the sink node is a , and a package from node e to node f follows the path $\langle e, c, a, d, f \rangle$.

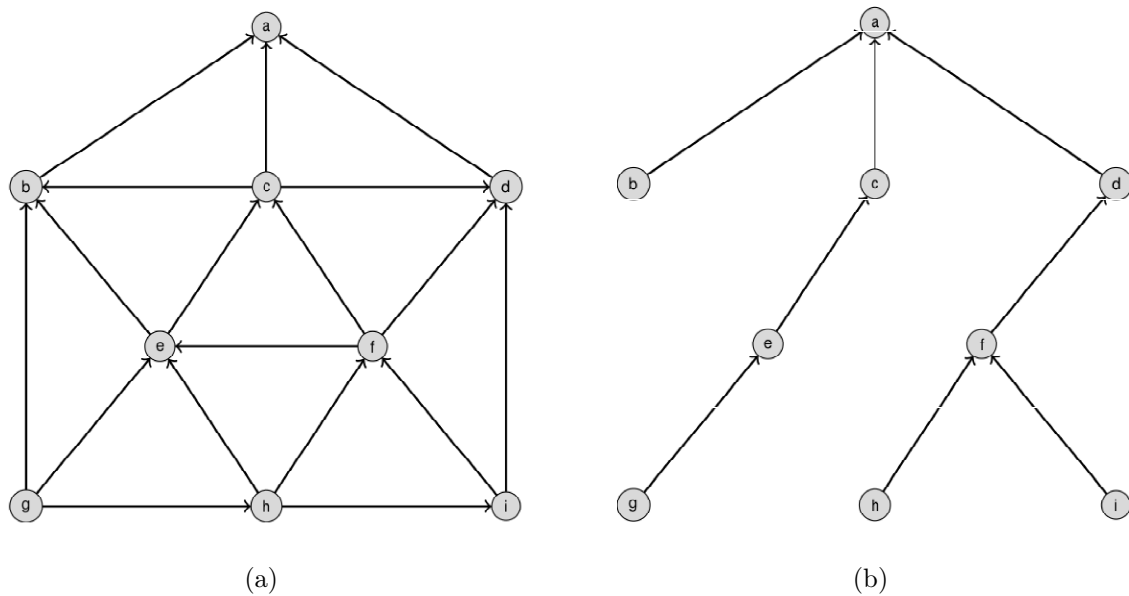


Figure 1.1. (a) An example of a DODAG rooted on node a , and (b) an example of a routing tree in this DODAG. In this example, a package from node e to node f follows the path $\langle e, c, a, d, f \rangle$

The most common metric used to build the RPL routing table is the link quality. RPL uses the last observation of the link quality to compute the routes. Therefore,

the performance of RPL may vary with the network links variability. The objective of this work is to improve the performance of RPL networks by taking into account link variability in the routing protocol.

Two main strategies can be used to optimize problems with data variability: stochastic programming [Spall, 2005] and robust optimization [Kouvelis and Yu, 1997; Ben-Tal and Nemirovski, 2002]. The former is mostly applied whenever the probability law associated to the uncertain data is known in advance. A drawback of this approach is that it is sometimes difficult to define the probability distribution associated to the uncertain data, or else errors can happen on the parameters estimation.

Robust optimization (RO) is an alternative to stochastic programming, where the variability at the uncertain data is represented by a set of deterministic values. The realization of the uncertain data, bounded by the determined interval, is called a scenario of a instance of a RO problem. RO is properly used when there is some uncertainly data that can be recovered from past times observation of the problem that will be optimized. A RO framework presented in [Kouvelis and Yu, 1997] defines three critical steps to build a robust model. The first step is to structure the uncertain data. It defines that the uncertain data can be structured by a discrete set of scenarios, each one with a single value for each uncertain data, or by an interval of values for each uncertain parameter, which leads to an infinite number of scenarios. The second step is to choose an appropriate robust criterion, that determines how conservative will be the robust model. The last step is to gather the chosen data structure and robust criterion to build a robust model. Given the robust model, a solution is said to be robust if it has the smallest value for the robust criterion, among all feasible solutions. The robust optimization problem is defined as to find the robust solution for a given robust model.

There are three main robust criteria, namely the *regret* (RE) [Kouvelis and Yu, 1997; Karaşan et al., 2001], the *minmax regret* (MR) [Kouvelis and Yu, 1997; Aissi et al., 2009], and the *minmax relative regret* (RR) [Kouvelis and Yu, 1997; Coco et al., 2014a]. They are explained below.

Let R be the set of possible scenarios for a given problem, and X be the set of feasible solutions for this problem. Also, let \bar{x}_r be the cost of solution x for the scenario r . The RE criterion is defined as

$$\min_{x \in X} \max_{r \in R} \{\bar{x}_r\}, \quad (1.1)$$

i.e., the robust solution is the one that minimizes the maximum value of \bar{x}_r over all possible scenarios. This criterion has a conservative nature, as it is based on the anticipation that the worst scenario might happen [Kouvelis and Yu, 1997].

Let R , X and \bar{x}_r be as defined above. Let also \bar{x}_r^* be the cost of the optimal solution x_r^* for the scenario r . The MR criterion is defined as

$$\min_{x \in X} \max_{r \in R} \{\bar{x}_r - \bar{x}_r^*\}, \quad (1.2)$$

i.e., the robust solution is one that minimizes the maximum regret over all scenarios. Analogously, the MRR criterion is defined as

$$\min_{x \in X} \max_{r \in R} \left\{ \frac{\bar{x}_r - \bar{x}_r^*}{\bar{x}_r^*} \right\}, \quad (1.3)$$

i.e., the regret of using x_r instead of x_r^* relative by the cost of x_r [Kouvelis and Yu, 1997]. MRR is more difficult to solve, when compared to MR, because its objective function is nonlinear. However, MRR is a better metric than MR, as shown in [Coco et al., 2014a].

In this work, the RPL routing problem is modeled as a robust optimization problem derived from the Shortest Path Tree problem (SPT) [Wu and Chao, 2004; Cormen et al., 2009]. Given a connected digraph $G = (V, A)$ with a set V of nodes and a set A of arcs. Each arc $(i, j) \in A$ is associated with a cost $c_{ij} \in \mathbb{R}_+$. Moreover, let $n = |V|$ and $m = |A|$ be respectively the number of nodes and arcs in G . SPT consists in finding a spanning tree of G that has the shortest path from a given root node s to every node in $V \setminus \{s\}$. There are polynomial time algorithms that solve the SPT, such as Dijkstra algorithm [Dijkstra, 1959] and Bellman-Ford algorithm [Bellman, 1956].

The Robust Shortest Path Tree problem (RSPT) is a generalization of SPT, where the cost of each arc $(i, j) \in A$ is defined by an interval $[l_{ij}, u_{ij}]$, with $l_{ij}, u_{ij} \in \mathbb{R}_+$, such that $u_{ij} \geq l_{ij} > 0, \forall (i, j) \in A$. Also, let $s \in V$ be the root node. In our model for the RPL routing problem, the nodes in V are associated to IoT devices, and the arcs in E are associated to links. Besides, for all $(i, j) \in A$, l_{ij} and u_{ij} correspond to the smallest and the largest observation in the link quality metric. This problem is formally defined in Chapter 4.

The remainder of this work is organized as follows. The routing problem in 6LoWPAN network is discussed in Chapter 2. Related works are presented in Chapter 3. The RSPT routing problem is formally defined in Chapter 4, and a mathematical formulation is presented. Heuristics for RSPT are proposed in Chapter 5. Computational experiments are presented in Chapter 6. Finally, concluding remarks are drawn in Chapter 7.

Chapter 2

Routing in 6LoWPAN

Embedded devices (ED), also called *smart objects*, are a part of the Internet that consists in resource-constrained IP-enabled devices [Shelby and Bormann, 2011; Ee et al., 2010]. The IoT connects such devices, and will impact significantly in various sectors of the society, as environmental monitoring, energy savings, smart grids, more efficient factories, logistics, healthcare and smart homes [Atzori et al., 2010; Giusto et al., 2010; Xia et al., 2012; Shelby and Bormann, 2011; Gubbi et al., 2013; Kushalnagar et al., 2007; Kortuem et al., 2010; Said and Masud, 2013; Madakam et al., 2015].

	smart office/building	smart city	smart agriculture	smart transportation
network size	small	medium	medium/large	large
# users	not too many	many, general public	few, landowners public	large, general
energy	battery	battery, harvesting	battery, harvesting	battery, harvesting
databases	local	shared	local, shared	shared
connectivity	wifi/3G/4G/backbone	wifi/3G/4G backbone	wifi/satellite	wifi/satellite
bandwidth	small	large	medium	medium/large
examples	Furdík and Lukác [2012]	Zanella et al. [2014]	TongKe [2013]	Lee et al. [2015]

Table 2.1. Smart environment application domains [Gubbi et al., 2013]

Table 2.1 gives details about some of these application domains. The first column shows the characteristic of the network. Remaining columns describe different application domains. Each type of network has its own characteristics. One can see that IoT networks can be a small building network, as presented by Furdík and Lukác [2012], or a great network that interconnect the transport systems in a city, with thousands

of EDs connecting the public transportation, as presented by Lee et al. [2015]. This work focus on office/building networks, with low bandwidth and hundred of devices.

The network constructed by interconnected wireless EDs is a subset of the IoT, and is called *Wireless Embedded Internet* (WEI). The 6LoWPAN was developed to enable WEI to work with the IPv6 protocol, instead the traditional IPv4 protocol, thus enabling a vast amount of dispositives to be on the IoT [Shelby and Bormann, 2011].

The EDs communication devices typically benefit from multi-hop mesh networks, i.e. networks wherein messages are routed from a source to a destination through possibly multiple other nodes acting as relays. A 6LoWPAN is a multi-hop mesh network, such that a path is formed by the set of links (*i.e.*, channels) between the source and the destination. However, current IPv4 protocols may not easily be applicable to these networks, as discussed in [Shelby and Bormann, 2011].

Some protocols were designed for 6LoWPAN networks [Winter, 2012; Ee et al., 2010; Vasseur et al., 2011; Felsche et al., 2012]. One of the first protocols is the 6LoWPAN On-demand Distance Vector Routing (LOAD) [Kim et al., 2007b; Chang et al., 2010]. It is a simplified on-demand routing protocol based on Ad-Hoc On-demand distance vector routing (AODV) [Royer and Perkins, 1999; Perkins et al., 2003], that was first proposed for wireless sensor networks (WSN). Both LOAD and AODV use IEEE 64-bit address as devices interface identifiers for building it routing table. However, because of its length, the IEEE address is not scalable and inefficient when used in a 6LoWPAN network [Zhu et al., 2009].

The Dynamic MANET On-demand for 6LoWPAN Routing protocol (DYMO-low) [Kim et al., 2007a] is based on the Dynamic MANET On-demand Routing protocol (DYMO) [Chakeres and Perkins, 2008] and the AODV protocol. DYMO-low is a routing protocol for 6LoWPAN that works directly on the link layer, not using the network IP layer. Thus, it creates a mesh network topology of 6LoWPAN devices that do not have the knowledge of IP, such that IP sees this network as a single link, with all EDs sharing a same IPv6 prefix. This protocol is able to use both 16-bit link layer short address or IEEE 64-bit extended address.

The Hierarchical Routing protocol (HiLow) [Kim et al., 2005] was proposed in order to increase the network scalability. It works with 16 bits link layer short address and it is characterized by low memory consumption and a fast packet routing [Ee et al., 2010]. Besides that, HiLow do not have a recovery path mechanism, making it unable to construct another routing table when some node of the network fails. This characteristic makes HiLow unable to work with dynamic networks. Therefore, it can efficiently route packets only in static environments.

The IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [Vasseur et al., 2011; Winter, 2012; Gaddour and Koubâa, 2013] was designed to be highly adaptive to network conditions and to provide alternate routes, whenever default routes are inaccessible. RPL also works with 16 bits link layer short address, one of the characteristics of 6LoWPAN.

RPL is the most promising routing protocol for 6LoWPAN when compared to other protocols [Gaddour and Koubâa, 2013]. RPL is flexible, supporting various types of traffic (MP2P, P2MP and P2P). Thus, it can be easily adapted for various application requirements. It is also flexible and can use any metric to build its routing table. Besides, RPL is suitable for fault-tolerance applications, with a local and a global repair of its routing table [Gaddour and Koubâa, 2013].

RPL is based on the topological concept of a DODAG. It is possible to define a network by one or more of this DODAGs at the same time, thus being denominated a RPL instance. Furthermore, one network can run multiple RPL instances at the same time, such that each instance serves to an different application. Thus, one node can join one or more of these instances [Gaddour and Koubâa, 2013].

The RPL protocol is composed of three different ICMPv6 control messages: (*i*) DIS (DODAG Information Solicitation); (*ii*) DIO (DODAG Information Object); (*iii*) DAO (DODAG Destination Advertisement Object) [Winter, 2012]. These messages are used to exchange information about the DODAG topology.

A comparison between RPL and the others 6LoWPAN routing protocols can be found in [Gaddour and Koubâa, 2013]. It shows that RPL is a flexible protocol that can directly connect to the Internet by using the IPv6 protocol, supports various types of traffic, and can be easily adapted for various types of applications. Therefore, [Gaddour and Koubâa, 2013] concludes that RPL outperform others 6LoWPAN routing protocols. A detailed description of how RPL works can be found in [Vasseur et al., 2011; Winter, 2012].

Values of link quality are used to make routing decisions. However, the great link variability in 6LoWPANs can degrade the link throughput, as observed in Koksâl et al. [2006]. If one link channel varies, it can disrupt the network and cause the whole network to lose performance, or even fail [Atzori et al., 2010; Shelby and Bormann, 2011; Koksâl et al., 2006]. Besides, no current routing protocols in the literature consider channel variability.

Routing algorithms for 6LoWPAN try to select the paths with better channels in order for messages to be transmitted more efficiently, consuming less energy and avoiding retransmissions. However, the channel variability may lead the routing selection to the worst possible decision in that instant, causing many problems in the network.

When this occurs, the network will fail, it may even get disconnected. Besides that, a large amount of energy will be used in order to retransmit messages that were lost in the network. Thus, the network will be even more congested due the high quantity of retransmitted messages.

This work propose a solution to deal with the channel variability. The routing problem of the RPL under channel variability is modeled as a RSPT. RO techniques are used to select the routes and improve the network, even in the presence of wireless channel variability, thus modeling the RPL routing protocol as a RSPT. This work aims to construct a DODAG in a given network by considering the best and the worst link quality in a given interval of time, and then build a routing tree that will be more resilient to this link quality variation.

Chapter 3

Related works

Robust optimization problems were first studied in [Gupta and Rosenhead, 1968]. Many RO problems were introduced by Kouvelis and Yu [1997]. For instance, the Robust Assignment problem [Deí et al., 2006; Pereira and Averbakh, 2011], the Robust Shortest Path problem [Karaşan et al., 2001; Coco et al., 2014a, 2016], the Robust Minimum Spanning Tree Problem [Yaman et al., 2001; Kasperski et al., 2012], the Robust Knapsack problem [Yu, 1996; Monaci and Pferschy, 2013] and Robust Network Design problem [Gutiérrez et al., 1996; Ukkusuri et al., 2007]. More recently, RO versions of classical \mathcal{NP} -Hard problems have also been studied, as the Robust Travelling Salesman problem [Montemanni et al., 2006; Candia-Véjar et al., 2011], the Robust Set Covering problem [Degel and Lutter, 2013; Coco et al., 2015], the Robust Scheduling Optimization problems [Daniels and Kouvelis, 1995; Wu et al., 2009], and the Robust Restricted Shortest Path problem [Assunção et al., 2014], among others. This work refers to the survey [Ben-Tal and Nemirovski, 2002] for other RO problems. It also presents the main results of RO applied to uncertain linear, conic quadratic and semidefinite programming until the time of its publication.

A survey about robust optimization criteria is presented by Coco et al. [2014b]. This work addresses RO problems with discrete scenarios. The three main *minmax* regret criteria are reviewed, so as α -robustness [Kalai et al., 2012], *bw*-robustness [Roy, 2010], and *pw*-robustness [Gabrel et al., 2013]. It also presents two new robust criteria that can be applied to measure the robustness of both scenarios and interval data models.

There are several works in literature that deals with the complexity theory of RO problems. The survey [Aissi et al., 2009] is dedicated to computational complexity results for several versions of Robust Shortest Path, Robust Minimal Spanning Tree, and Robust Knapsack. The book [Kouvelis and Yu, 1997] also gives complexity results

for a widely range of RO problems. The work of Chekuri et al. [2007] proves that a single-source version of the Robust Network Design problem is \mathcal{NP} -Hard in undirected graphs. The complexity of the Robust Least-Squares problem have be discussed in [Ghaoui and Lebre, 1997]. The work [Aron and Hentenyck, 2004] prove that the Robust Minimal Spanning Tree problem is \mathcal{NP} -Complete even in graphs when the edges cost are defined as binary values, settling the conjecture presented in [Kouvelis and Yu, 1997]. The work of Averbakh [2001] presents the first problem that is \mathcal{NP} -Hard when there is a finite set of scenarios, but is polynomial time solvable when the uncertainty is represented by interval data.

The first MILP formulation for the Robust Shortest Path problem was given by Karaşan et al. [2001]. Besides, a pre-processing algorithm for the Robust Shortest Path problem was given in this same work. It consists in finding the so called *weak arcs*, i.e. arcs that never will be in the optimal solution of the problem. This algorithm is valid only for acyclic, planar or layered graphs. The idea of identifying weak arcs in a solution is extended to the Robust Travelling Salesman problem in [Montemanni et al., 2006, 2007].

A constraint programming approach to the Robust Minimal Spanning Tree problem with interval data was developed by Aron and Hentenyck [2002]. The proposed search algorithm is based on three basic components: a combinatorial lower bound, a pruning component and a branching heuristic. Computational experiments show that the proposed algorithm outperforms the MILP based branch-and-bound algorithm proposed by Yaman et al. [2001].

Benders decomposition approaches for the Robust Shortest Path problem and the Robust Minimal Spanning Tree problem are presented by Montemanni and Gambardella [2005a] and Montemanni [2006], respectively. These works extend the previous works of Montemanni et al. [2004] and Montemanni and Gambardella [2005a], that have presented branch-and-bound algorithms for these two problems. It is shown that the benders decomposition approach outperforms the previous branch-and-bound algorithms and all others state-of-the-art algorithms until the time of its publication.

The work of Kouvelis and Yu [1997] is extended in Averbakh [2005]. It considers the robust version of combinatorial optimization problems with discrete scenarios and the RR criterion. The author proposes a nonlinear generic formulation for this class of problem, and also developed a proof that RO problems with interval data with the RR criterion are \mathcal{NP} -Hard.

A 2-approximative algorithm for RO problems with interval data and the RR criterion is presented by Kasperski and Zieliński [2006]. This algorithm fixes the mean scenario, i.e. a scenario where each arc cost is fixed at its mean value $((l_{ij} + u_{ij})/2)$, and

then runs an algorithm for the classical version of the problem. This 2-approximative algorithm has the same complexity as the algorithm for the classical problem.

A polynomial time $(1 + \varepsilon)$ -approximative scheme for RO problems is presented by Kasperski and Zieliński [2007], extending the work [Kasperski and Zieliński, 2006]. This algorithm can be applied for all RO problems that its classical version is polynomially solvable. Its complexity for the Robust Shortest Path problem is $\mathcal{O}(|A|^3/\varepsilon^2)$, where A is the set of arcs of the problem.

Approximative algorithms for RO problems with interval data and AR criterion was developed in Conde [2012]. This work extends the previous work of Kasperski and Zieliński [2006, 2007]. It aims to solve the classical version of RO problems at the mean scenario, obtaining algorithms that have an approximation factor of 2 for *minmax* absolute regret problems [Kasperski and Zieliński, 2006, 2007].

Four pre-processing algorithms for Robust Shortest Path were developed by Catanzaro et al. [2011]. This work extends the work of Karaşan et al. [2001], by presenting new algorithms to fix arcs in the optimal solution. The first algorithm aims to find an subset of arcs that, if removed from the problem, result in solutions with a worst robustness cost. The second algorithm search for nodes that are not in any path between the origin node s and destination node t . The third algorithm construct a Minimum Spanning Tree (MST) t at the upper scenario, i.e. a scenario where each arc of the graph is at it respective maximum value, and then find a shortest path p^1 between s and t in t . Then, one arc $(i, j) \in p^1$ is removed from the original graph and a new shortest path p^2 is evaluated at the resultant graph. If the robustness cost of shortest path p^1 is lower than shortest path p^2 , then the this arc (i, j) belongs to the optimal solution of the problem. The last algorithm construct a MST t' at the lower scenario, i.e. a scenario where each arc of the graph is at it respective minimum value, and then find a shortest path p'^1 between s and t in t' . After that, an arc $(i, j) \in G \setminus t'$ is inserted in t' , and a new shortest path p'^2 is evaluated in t' . If this new arc (i, j) do not belongs to the shortest path p'^2 , then it can be removed from the problem. These four algorithms has complexity equals $\mathcal{O}(|V|^2 \times |A|)$, $\mathcal{O}(|V|^3)$, $\mathcal{O}(|V|^2 \times |A|)$, and $\mathcal{O}(|V|^2 \times |A|)$, respectively, where V represents the set of nodes and A represents the set of arcs of a graph G .

A Scenario-based Algorithm (SBA) for the Robust Set Covering problem with interval data was presented by Coco et al. [2015]. This work extends the approximative algorithms presented by Kasperski and Zieliński [2006], targeting a set of scenarios computed as a linear combination of mean and upper scenarios. The presented heuristic is also a 2-approximative algorithm, as it also includes the mean scenario. It is shown that SBA outperforms the heuristics presented by Kasperski and Zieliński [2006] in a

reasonable running time. An extension of this work is presented in Coco et al. [2016], that develops hybrid algorithms that combine SBA and exact methods for this same problem.

For the best of our knowledge, there is no work in literature for RSPT. Thus, this work proposes the first MILP formulation for this problem, so as exact, approximative, and heuristic algorithms.

Chapter 4

The Robust Shortest Path Tree

4.1 The problem

Let $G = (V, A)$ be a connected digraph, where V is the set of nodes and A is the set of arcs. Each arc $(i, j) \in A$ is associated with an interval cost $[l_{ij}, u_{ij}]$, which represents the variation in the link quality metric, with $l_{ij}, u_{ij} \in \mathbb{R}_+$ and $u_{ij} \geq l_{ij} > 0$, for all arcs $(i, j) \in A$. Let also $s \in V$ be the root node, and $V' = V \setminus \{s\}$.

Definition 4.1.1. A scenario r is a realization of arcs cost $c_{ij}^r \in [l_{ij}, u_{ij}]$ for each arc $(i, j) \in A$.

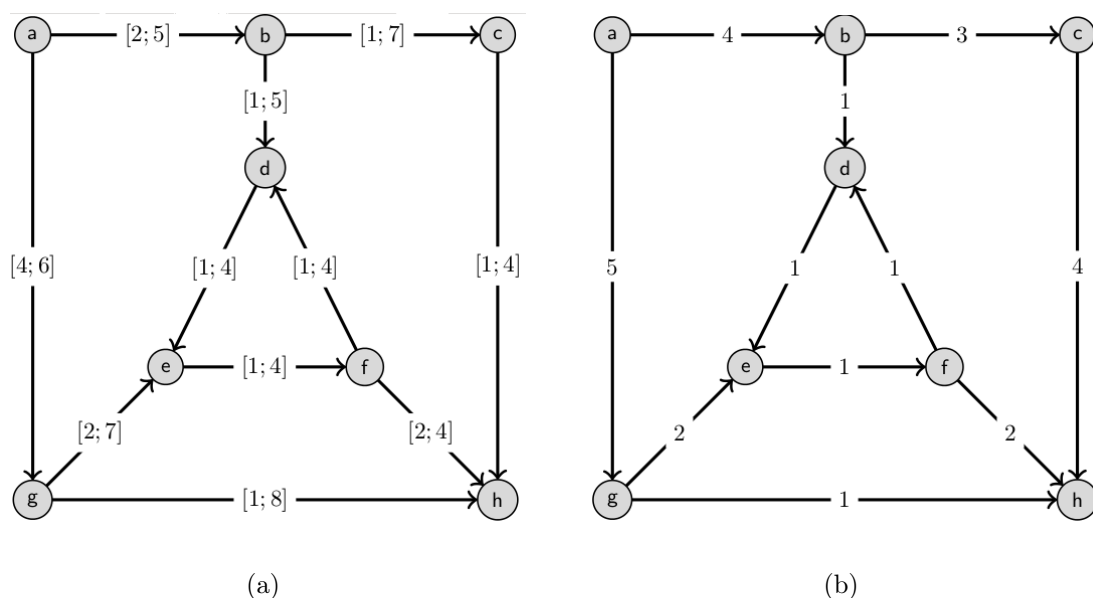


Figure 4.1. Examples of a RSPT instance (a) and one possible scenario (b)

Figure 4.1 shows an example of a RSPT instance and one possible scenario defined in this network. One can see that Figure 4.1b has a cost for each arc defined inside the interval cost presented in Figure 4.1a.

Let \mathcal{T} be the set of all s -rooted spanning trees in G . Let also p_i^t be the path from node s to node $i \in V'$ induced by the tree $t \in \mathcal{T}$. Besides, let $c^r(p_i^t) = \sum_{(i,j) \in A[p_i^t]} c_{ij}^r$ be the cost of p_i^t in the scenario r , and $c^r(p_i^*)$ be the cost in the scenario r of the shortest path p_i^* from s to i in r .

Definition 4.1.2. The *robust deviation* of a path p_i^t induced by $t \in \mathcal{T}$ in the scenario r (also referred to as the *regret* of p_i^t in r) is defined as $d^r(p_i^t) = c^r(p_i^t) - c^r(p_i^*)$, i.e. the difference between the cost of p_i^t in r and the best possible path from s to i in r .

Let $A[t]$ be the set of arcs that compose a tree $t \in \mathcal{T}$, and $A[p_i^t] \subset A[t]$ be the set of arcs that compose the path p_i^t , and \mathcal{R} be the set of all possible scenarios in G .

Lemma 4.1.1. [Kouvelis and Yu, 1997] *The robust deviation of p_i^t is the maximum, among all scenarios in \mathcal{R} , at the scenario $r_i^t \in \mathcal{R}$, such that $c_{ij}^{r_i^t} = 1$, for all $(i, j) \in A[p_i^t]$, and $c_{ij}^{r_i^t} = 0$, for all $(i, j) \in A \setminus A[p_i^t]$.*

The smaller is the robust deviation of p_i^t , the better is this path for sending packets from s to i . Besides, the smaller is the value of $d^{r_i^t}(p_i^t)$, the more robust is this path to variations in the link quality.

Definition 4.1.3. The *robustness cost* of $t \in \mathcal{T}$ is defined as $R_t = \sum_{i \in V'} d^{r_i^t}(p_i^t)$, i.e. the sum of the maximum robust deviation of every path from s to any other node in G .

A smaller robustness cost implies that, when there is a great link variability, the network will be less affected. The smallest is robustness cost of the communication tree of a 6LoWPAN, the more efficient and the more reliable is the network.

Definition 4.1.4. A tree $t^* \in \mathcal{T}$ is said to be *robust* if it has the smallest robustness cost among all trees in \mathcal{T} .

Therefore, RSPT can be defined as to find a robust spanning tree of G rooted in s , i.e. $t^* = \arg \min_{t \in \mathcal{T}} R_t$.

4.1.1 RSPT complexity proof

In order to proof that RSPT is \mathcal{NP} -Complete, it is possible to construct reductions from two well known \mathcal{NP} -Complete problems: the *2-Partition Problem* (2PT) [Karp,

1972] and the *3-Partition Problem* (3PT) [Garey and Johnson, 1979]. It is proved that, with only two different scenarios, the RSPT problem is \mathcal{NP} -Complete, and with an undefined number of scenarios, the RSPT problem is strongly \mathcal{NP} -Complete.

Definition 4.1.5. Given a finite set I , and a weight $a_i \geq 0, \forall i \in I$, the objective of the *2-Partition Problem* is to find two subsets of I , namely X and Y , such that $\sum_{i \in X} a_i = \sum_{i \in I \setminus Y} a_i$ and $X \cap Y = \emptyset$. Karp [1972] demonstrated that 2PT is \mathcal{NP} -Complete even when $|X| = |Y| = \frac{|I|}{2}$.

Definition 4.1.6. Given a finite set I of exactly $3l$ elements, and a weight $a_i \geq 0, \forall i \in I$, the objective of the *3-Partition Problem* is to find m disjoint subsets $s_1, s_2, \dots, s_m \subseteq I$ such that $s_1 \cup s_2 \cup \dots \cup s_m = I$, and $\sum_{k \in s_i} a_k = B, \forall s_i \in m$. Garey and Johnson [1979] demonstrated that 3PT is strongly \mathcal{NP} -Hard even when $\frac{B}{4} < a_i < \frac{B}{2}, \forall a_i \in I$. It is possible to note that, when this inequality is satisfied, a solution for 3PT consists in exactly $m = l$ triplets such that the sum of elements in each triplet is B .

Theorem 4.1.2. *The RSPT problem is \mathcal{NP} -Complete, even in layered graphs with only 3 layers and 2 scenarios.*

Proof. The 2PT problem can be reduced to RSPT. Given a 2PT instance, it is possible to construct a corresponding RSPT instance in polynomial time.

Let I be a finite set with $|I| = m$ elements. It is possible to build a digraph $G = (V, A)$ partitionated into 3 disjoint subsets of vertices $V = V_0 \cup V_1 \cup V_2$. Subset V_0 contains only one vertice, denominated s . Subset V_1 contains $2m$ vertices, and subset V_2 contains m vertices. Besides, it is possible to define three disjoint subsets of arcs $A = A_0 \cup A_1 \cup A_2$. Subset A_0 contains $2m$ arcs, constructing an complete bipartite digraph between subsets V_0 and V_1 . Subset A_1 contains m arcs, such that there is an arc from each vertice $v_i \in V_1$ to a vertice $w_i \in V_2$, for all $0 \leq i < m$. One can see that this construction corresponds to a layered digraph with 3 layers of vertices.

A RSPT problem can be constructed with only two scenarios, namely r_1 and r_2 . The cost of arcs $e \in A_0$ are fixed in 0 under both scenarios. At scenario r_1 , the cost of each arc $e_i \in A_1$ is fixed in a_i , and the cost of arcs $e \in A_2$ is fixed in 0. At scenario r_2 , the cost of each arc $e_i \in A_2$ is fixed in a_i , and the cost of arcs $e \in A_1$ is fixed in 0.

Given the above construction, let t_1 and t_2 be two shortest paths trees (SPT) in G with root at vertice s in scenarios r_1 and r_2 , respectively. Note that t_1 and t_2 are unique by construction. Besides, let X and Y be two set of arcs such that $X = E \setminus A[t_2]$ and $Y = E \setminus A[t_1]$. A 2PT of I exists if and only if $R_{t_1} = R_{t_2} = \frac{1}{2} \sum_{k \in I} a_k$. To prove the "if" part, suppose there exists a partition of I in two subsets X and Y with $\sum_{k \in X} a_k = \sum_{k \in Y} a_k = \frac{1}{2} \sum_{k \in I} a_k$. It is possible to construct two SPT t_1 and t_2 such

that $A[t_1] = E_0 \cup X$ and $A[t_2] = E_0 \cup Y$. Both t_1 and t_2 are optimal solutions for the RSPT instance. To prove the "only if" part, assume that $R_{t^*} = \frac{1}{2} \sum_{k \in I} a_k$. Let t_1 and t_2 be two SPT at scenarios r_1 and r_2 , respectively. Moreover, let X and Y be two subsets of arcs such that $X = A \setminus A[t_2]$ and $Y = A \setminus A[t_1]$. By the construction of the network, it is clear that $(A[t_1] \cap A[t_2]) \setminus E_0 = \emptyset$. Also, by the construction of the network, it is clear that $R_{t_1} = R_{t_2} = \frac{1}{2} \sum_{k \in I} a_k$. Thus, it must be a 2PT of I in subsets X and Y . \square

Theorem 4.1.3. *The RSPT problem is strongly \mathcal{NP} -Complete for an unbounded number of scenarios, even in layered graphs with only 3 layers.*

Proof. The 3PT problem can be reduced to RSPT. Given a 3PT instance, it is possible to construct a corresponding RSPT instance in polynomial time.

Let I be a finite set with $|I| = 3l$ elements. It is possible to build a digraph $G = (V, A)$ partitionated into 3 disjunt subsets of vertices $V = V_0 \cup V_1 \cup V_2$. Subset V_0 contains only one vertice, denominated s . Subset V_1 contains $(3l)^2$ vertices, and subset V_2 contains $3l$ vertices. Besides, it is possible to define $l + 1$ disjoint subsets of arcs $A = A_1 \cup A_2 \cup \dots \cup A_l \cup A_{l+1}$. Subset A_{l+1} contains $(3l)^2$ arcs, constructing an complete bipartite digraph between subsets V_0 and V_1 . All others l subsets contain $3l$ arcs with origin in vertices $v_{3l+i} \in V_1$ and destiny at vertices $w_i \in V_2$, for $i \in \{0, 1, \dots, 3l\}$. One can see that this construction corresponds to a layered digraph with 3 layers of vertices.

A RSPT problem can be constructed with an unbounded number l of scenarios. The cost of arcs $e \in A_{l+1}$ are fixed at 0 under all scenarios. For each scenario $r_i \in \{0, 1, \dots, l\}$, the cost of each arc $e \in A_i$ is fixed in a_i . Besides, the cost of each arc $e \in A \setminus A_i$ is fixed in 0.

Given the above construction, let $\mathcal{T} = \{t_1, t_2, \dots, t_l\}$ be the set of SPT of G with root at vertice s in G on scenarios $\{r_1, r_2, \dots, r_l\}$, respectively, such that $A[t_i] = E_0 \setminus \bigcup_{j \in l, j \neq i} t_j$. Note that all SPT $\in \mathcal{T}$ are unique by construction. A 3PT of I exists if and only if there is a RSPT t^* of G with $R_{t^*} = B$. To prove the "if" part, suppose there exists a 3PT of I into l disjoint subsets with $\sum_{k \in s_m} a_k = B, \forall m \in \{1, 2, \dots, l\}$. It is possible to construct a set $\{t_1, t_2, \dots, t_l\}$ of SPTs in G as mentioned above. By the construction of the network, the robustness cost of all SPTs are the same, i.e. $R_{t_i} = B, \forall t_i \in \mathcal{T}$. Thus, $R_{t^*} = \min R_{t_i}, \forall t_i \in \mathcal{T}$, that, by definition, is equal to B . To prove the "only if" part, assume that $R_{t^*} = B$. By the construction of the network, there are l different SPT in G , each one on a different scenario. Besides, all l SPT have the same robustness cost B . It is possible to define l disjoint subsets of arcs s_1, s_2, \dots, s_l , such that the subset s_i contains the arcs in $t_i \cap \bigcup_{j \in l, j \neq i} t_j$. By definition,

$\sum_{k \in I} a_k = lB$. As there are l SPT in G with robustness cost equal to B , there is exactly a 3PT of I into subsets s_1, s_2, \dots, s_l . \square

4.2 A mixed integer linear programming formulation

A Mixed Integer Linear Programming (MILP) formulation is defined with decision variables $z_{ij} \in \{0, 1\}$ such that $z_{ij} = 1$ if arc $(i, j) \in A$ belongs to the robust tree and $z_{ij} = 0$ otherwise. Moreover, auxiliary variables $y_{ij}^k \in \{0, 1\}$ keep the path from s to $k \in V \setminus \{s\}$ induced by the tree defined by variables z_{ij} . Besides, variables $x_i^k \geq 0$ have the cost of the shortest path from s to $i \in V \setminus \{s\}$ in the scenario induced by the path from s to k (defined by variables y_{ij}^k). Therefore, the shortest path from s to k in this scenario is kept in x_k^k . The corresponding MILP formulation is defined by Equations (4.1) to (4.9).

$$\min \sum_{k \in V \setminus \{s\}} \left[\sum_{(i,j) \in A} (u_{ij} y_{ij}^k) - x_k^k \right] \quad (4.1)$$

$$\sum_{(j,l) \in A} y_{jl}^k - \sum_{(i,j) \in A} y_{ij}^k = \begin{cases} 1, & \text{if } j = s \\ -1, & \text{if } j = k \\ 0, & \text{otherwise} \end{cases}, \quad \forall j \in V, k \in V \setminus \{s\} \quad (4.2)$$

$$x_j^k \leq x_i^k + l_{ij} + (u_{ij} - l_{ij}) y_{ij}^k, \quad \forall (i, j) \in A, k \in V \setminus \{s\} \quad (4.3)$$

$$y_{ij}^k \leq z_{ij}, \quad \forall (i, j) \in A, k \in V \setminus \{s\} \quad (4.4)$$

$$\sum_{(i,j) \in A} z_{ij} = n - 1 \quad (4.5)$$

$$x_s^k = 0, \quad k \in V \quad (4.6)$$

$$x_i^k \geq 0, \quad \forall i \in V \setminus \{s\}, k \in V \quad (4.7)$$

$$y_{ij}^k \in \{0, 1\}, \quad \forall (i, j) \in A, k \in V \quad (4.8)$$

$$z_{ij} \in \{0, 1\}, \quad \forall (i, j) \in A \tag{4.9}$$

The objective function (4.1) minimizes the sum of the maximum robust deviation for every $k \in V \setminus \{s\}$. Constraints (4.2) are the classic flow conservation constraints and ensure the connectivity of each path from s to k . Inequalities (4.3) enforce the correct value of x . The cost of the shortest path from vertex s to a vertex $j \in V \setminus \{s\}$ is computed as the cost of the path from s to a vertex i that is adjacent to j and belongs to its inverse transitive closure, plus u_{ij} if $y_{ijk} = 1$ or l_{ij} otherwise. Inequalities (4.4) guarantee that the path from the root node s to every other node to be at the robust tree. Constraints (4.2), (4.4), and (4.5) enforce that variables z induces a tree. Equation (4.6) set to zero the value of the shortest path from s to itself, for all $k \in V \setminus \{s\}$. The domain of the variables x , y , and z are defined by (4.7), (4.8), and (4.9) respectively.

Chapter 5

Heuristics for RSPT

Kasperski and Zieliński [2006] developed three heuristics for the Robust Shortest Path problem. They are called Average Median (AM), Average Upper (AU) and Average Median Upper (AMU). This work first extends the AM, AU, and AMU heuristics for the RSPT.

AM receives as input a graph $G = (V, A)$, the lower bound l_{ij} , and the upper bound u_{ij} for each arc $(i, j) \in A$. Next, the scenario r^m is fixed, where the cost of the arcs are set to their respective mean value, i.e. $c_{ij}^m = (l_{ij} + u_{ij})/2$. Then, the regret of the solution is computed by constructing two Shortest Path Tree (SPT) by the Dijkstra's algorithm. The first evaluates the cost of the SPT in scenario r^m . The second evaluates the cost of the SPT on the worst scenario, as stated at Lemma 4.1.1. Then, the regret is computed and returned. The asymptotic worst case complexity of this heuristic is the same as that of Dijkstra's algorithm, i.e. $\mathcal{O}(|E| + |V| \cdot \log(|V|))$. This heuristic is a 2-approximative algorithm for Robust Shortest Path. A proof that it is also 2-approximative for RSPT is given in Session 5.1.

AU receives the same input as AM. Next, the scenario r^u is fixed, where the cost of the arcs are in their respective maximum value, i.e. $c_{ij}^u = u_{ij}$. Then, the regret is computed in the same way as in AM and returned. The asymptotic worst case complexity of this heuristic is the same complexity of as that of AM.

AMU runs AM and AU and then returns the best solution found by these heuristics. As AMU uses AM, its also a 2-approximation algorithm for the RSPT.

A Scenario-based Algorithm (SBA) for the Robust Set Covering problem was presented in Coco et al. [2015, 2016], and is extended here to RSPT. SBA is an extension of AMU, where target scenarios between the lower scenario (r^l , a scenario where the cost of the arcs are set to their respective lower, i.e. $c_{ij}^m = l_{ij}$) and the upper scenario (r^u) are investigated. SBA have three different parameters: the initial scenario α ;

the final scenario β ; and the step size γ . All parameters have real values into the interval $[0, 1]$. Target scenarios are computed as $\alpha + \delta\gamma$, for all $\delta \in \{0, \dots, i\}$ such that $\alpha + \delta\gamma \leq \beta$. Thus, SBA investigates $\frac{\beta - \alpha}{\gamma}$ different scenarios. One can see that both mean scenario r^m and upper scenario r^u can be obtained with SBA. Thus, it can produce solutions at least as good as AMU and also holds an approximation ratio of at most 2 for the RSPT.

This work advances the literature by developing a new heuristic to solve the RSPT, denominated MILP-based Variable Neighbourhood Descent (MILP-VND), and it is presented in Section 5.2. MILP-VND is compared with AM, AU, AMU [Kasperski and Zieliński, 2006, 2007], SBA [Coco et al., 2015, 2016], and the MILP formulation (4.1)-(4.9) in Chapter 6.

5.1 Proof of AM 2-approximation for RSPT

Lemma 5.1.1. *Kasperski and Zieliński [2006] Let $t', t'' \in \mathcal{T}$ be two trees rooted in s . Then, the following inequality holds for all $r \in \mathcal{R}$:*

$$d^r(p_k^{t'}) \geq \sum_{(i,j) \in A[p_k^{t'}] \setminus A[p_k^{t''}]} u_{ij} + \sum_{(i,j) \in A[p_k^{t''}] \setminus A[p_k^{t'}]} l_{ij}, \quad \forall k \in V \setminus \{s\} \quad (5.1)$$

Lemma 5.1.2. *Kasperski and Zieliński [2006] Let $t', t'' \in \mathcal{T}$ be two trees rooted in s . Then, the following inequality holds:*

$$d^r(p_k^{t'}) \leq d^r(p_k^{t''}) + \sum_{(i,j) \in A[p_k^{t'}] \setminus A[p_k^{t''}]} u_{ij} + \sum_{(i,j) \in A[p_k^{t''}] \setminus A[p_k^{t'}]} l_{ij}, \quad \forall k \in V \setminus \{s\} \quad (5.2)$$

Theorem 5.1.3. *The performance ratio of algorithm AM for the RSPT is at most 2.*

Proof. Let $\bar{t} \in \mathcal{T}$ be the tree given by AM algorithm. Also, for all $k \in V \setminus \{s\}$, let $A[p_k^{\bar{t}}]$ be the set of arcs from vertex s to k in \bar{t} . We will prove that, for all $t \in \mathcal{T}$, $\sum_{k \in V \setminus \{s\}} d^r(p_k^{\bar{t}}) \leq 2 \sum_{k \in V \setminus \{s\}} d^r(p_k^t)$, for all $t \in \mathcal{T}, r \in \mathcal{R}$. It follows that

$$\sum_{k \in V \setminus \{s\}} \left(\sum_{(i,j) \in A[p_k^t] \setminus A[p_k^{\bar{t}}]} u_{ij} - \sum_{(i,j) \in A[p_k^{\bar{t}}] \setminus A[p_k^t]} l_{ij} \right) \geq \quad (5.3)$$

$$\sum_{k \in V \setminus \{s\}} \left(\sum_{(i,j) \in A[p_k^{\bar{t}}] \setminus A[p_k^t]} u_{ij} - \sum_{(i,j) \in A[p_k^t] \setminus A[p_k^{\bar{t}}]} l_{ij} \right)$$

From inequality (5.2), we have

$$d^r(p_k^{\bar{t}}) \leq d^r(p_k^t) + \sum_{(i,j) \in A[p_k^{\bar{t}}] \setminus A[p_k^t]} u_{ij} + \sum_{(i,j) \in A[p_k^t] \setminus A[p_k^{\bar{t}}]} l_{ij}, \quad \forall k \in V \setminus \{s\} \quad (5.4)$$

We can rewrite inequality (5.4) as

$$\sum_{k \in V \setminus \{s\}} d^r(p_k^{\bar{t}}) \leq \sum_{k \in V \setminus \{s\}} d^r(p_k^t) + \sum_{k \in V \setminus \{s\}} \left(\sum_{(i,j) \in A[p_k^{\bar{t}}] \setminus A[p_k^t]} u_{ij} + \sum_{(i,j) \in A[p_k^t] \setminus A[p_k^{\bar{t}}]} l_{ij} \right) \quad (5.5)$$

Inequality (5.1) together with inequality (5.3) yield

$$d^r(p_k^t) \geq \sum_{(i,j) \in A[p_k^t] \setminus A[p_k^{\bar{t}}]} u_{ij} - \sum_{(i,j) \in A[p_k^{\bar{t}}] \setminus A[p_k^t]} l_{ij} \geq \quad (5.6)$$

$$\sum_{(i,j) \in A[p_k^{\bar{t}}] \setminus A[p_k^t]} u_{ij} - \sum_{(i,j) \in A[p_k^t] \setminus A[p_k^{\bar{t}}]} l_{ij}, \quad \forall k \in V \setminus \{s\}$$

We can rewrite inequality (5.6) as

$$\begin{aligned}
& \sum_{k \in V \setminus \{s\}} d^r(p_k^t) \geq \tag{5.7} \\
& \sum_{k \in V \setminus \{s\}} \left(\sum_{(i,j) \in A[p_k^t] \setminus A[p_k^{\bar{t}}]} u_{ij} - \sum_{(i,j) \in A[p_k^{\bar{t}}] \setminus A[p_k^t]} l_{ij} \right) \geq \\
& \sum_{k \in V \setminus \{s\}} \left(\sum_{(i,j) \in A[p_k^{\bar{t}}] \setminus A[p_k^t]} u_{ij} - \sum_{(i,j) \in A[p_k^t] \setminus A[p_k^{\bar{t}}]} l_{ij} \right)
\end{aligned}$$

Inequalities (5.5) and (5.7) imply that $\sum_{k \in V \setminus \{s\}} d^r(p_k^{\bar{t}}) \leq 2 \sum_{k \in V \setminus \{s\}} d^r(p_k^t)$ for any $t \in \mathcal{T}$. Therefore, it is also valid for the robust tree, which completes the proof. \square

5.2 MILP-based Variable Neighbourhood Descent

The MILP-VND heuristic consists in a local search procedure that uses CPLEX to evaluate the neighbourhood of a solution. It implements a Variable Neighbourhood Descent [Talbi, 2009], a traditional local search metaheuristic that exploits the idea of neighbourhood change to escape from local optimum solutions [Hansen and Mladenović, 2001].

The MILP-VND uses the σ -opt neighbourhood structure [Lin, 1965], with $\sigma \in \{1, \dots, 4\}$. It receives as input a graph $G = (V, A)$, besides the lower bound l_{ij} and the upper bound u_{ij} for each arc $(i, j) \in A$. Then, it runs SBA and use its solution as an initial point for the local search procedure. As MILP-VND starts from the SBA solution, it is also returns a solution that worst cost it at most twice of the optimal one.

The use of CPLEX for solving the neighbourhood structure instead of the traditional enumeration strategy is justified by the complexity of verify the value of a neighbourhood. The evaluation of one solution consist in running two times the Dijkstra's algorithm, that is well-known to has the complexity of $\mathcal{O}(|A| + |V| \log |V|)$. As MILP-VND uses up to a 4-opt neighbourhood structure, the complexity to verify a complete neighbourhood using the enumeration strategy is up to $\mathcal{O}(|A| + |V| \log |V|)^4$. Thus, using CPLEX can speed up the local search procedure.

MILP-VND uses the constraint (5.8) to define a neighbourhood. It limits the search space by only considering solutions that differ exactly 2σ arcs from a given solution $t \in \mathcal{T}$.

$$\sum_{(i,j) \in t} z_{i,j} = |V| - 1 - \sigma \quad (5.8)$$

MILP-VND starts with the solution t obtained from SBA and the value of the regret of t . Next, it runs the local search procedure until the limit value of σ is reached. Each neighbourhood structure evaluation is done by CPLEX by inserting the Constraint (5.8) at the MILP formulation defined by Equations (4.1)-(4.9). If CPLEX finds a solution with a better regret than the current solution t , then t is replaced by the new generated solution, plus setting σ to its initial value. Otherwise, the value of σ is increased, in order to explore a larger neighbourhood. Finally, it removes the inserted constraint before evaluating a new neighbourhood. After the exploration of all neighbourhood structures without an solution improvement, it returns the current solution t and MILP-VND is ended.

Chapter 6

Computational Experiments

Computational experiments were performed on an Intel Xeon CPU E5645 with 2.4 GHz clock and 32 GB of RAM memory, running Linux operating system. The *branch-and-bound* implementation of the ILOG CPLEX version 12.6 with default parameter settings has been used to solve the MILP formulation (4.1)-(4.9). The heuristics have been implemented from scratch in C++ and compiled with GNU GCC version 5.2.1. The running time of CPLEX branch-and-bound was limited 24 hours.

The performance of AM, AU, AMU, SBA, and MILP-VND are assessed on a classic set of RSP instances, based on Karařan graphs [Karařan et al., 2001] with 100 and 200 nodes. Karařan graphs rely on a layered [Sugiyama et al., 1981] and acyclic [Bondy and Murty, 1976] topology, see for example Figure 6.1. Instances are generated as in Coco et al. [2014a]. The instance names are presented as K-n-a-b-c-d, where n is the number of nodes and d is the graph width. Parameters a and b are used to generate the intervals $[l_{ij}; u_{ij}]$. For each arc $(i, j) \in A$, a value $c \in [1; a]$ is randomly generated using a uniform distribution. Then l_{ij} and u_{ij} are randomly generated in $[(1 - b)c; (1 + b)c]$ and $[l_{ij}; (1 + b)c]$ respectively. M layers are considered, each one with d nodes. Thus $n = Md$. Each pair of consecutive layers m and $m + 1$, $m \in \{1, \dots, M - 1\}$, define a complete bipartite digraph: each node $i \in m$ is connected to each node $j \in m + 1$ by an arc. Parameter c refers only the Id of the instance. Moreover, an additional source node s is connected to each node in layer 1 by an arc and an additional sink node t receives an arc from every node in layer M , as shown in Figure 6.1. Karařan graphs are used for this work because they are already used for RSP problems [Coco et al., 2014a; Karařan et al., 2001; Montemanni and Gambardella, 2005b; Montemanni et al., 2004; Pérez et al., 2012]. Besides, they represent DODAGs, the same type of graph given by the RPL protocol.

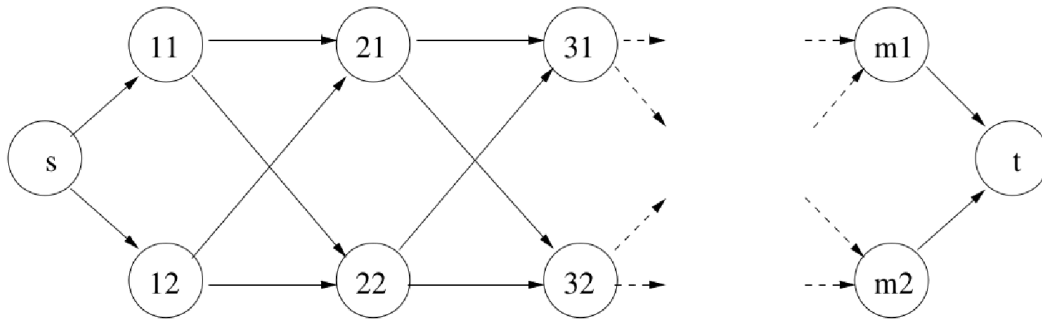


Figure 6.1. A Karagan graph with $M = m$ layers and width $W = 2$ [Karagan et al., 2001]

6.1 CPLEX branch-and-bound

The first experiment aims at evaluating the performance of the CPLEX branch-and-bound based on the MILP formulation (4.1)-(4.9) for the proposed instances. Besides, these results are used to evaluate the proposed heuristics in comparison to CPLEX upper bound.

The results are reported in Tables 6.1 and 6.2. The first column is the instances' name. The second and third columns show the lower bound and the upper bound founds by CPLEX, respectively. The fourth column shows the relative optimality gap. The last column shows the running time in seconds. It can be seen that CPLEX branch-and-bound easily solve at optimality all the instances with 100 nodes, with an maximum running time of 7.55 seconds. However, for the instances with 200 nodes, the CPLEX branch-and-bound can only solve four out of the ten evaluated instances in less than 24 hours. However, the maximum optimality gap observed for this set of instances was 1.56%.

6.2 Average Median Upper algorithm

The second experiment aims to evaluate AM, AU and AMU for the proposed instances. It compares the heuristics result with CPLEX branch-and-bound results presented in Tables 6.2 and 6.2. The results of these experiments are shown in Tables 6.3 and 6.4. The first column is the instances' name. The second and third columns report CPLEX branch-and-bound gap and running time. Then, each pair of columns refer to a heuristic. The first column of each pair shows the relative gap of the solution found by the heuristic over the upper bound of the CPLEX branch-and-bound, in percentage. The second column of each pair shows the heuristic running time in seconds.

instance	lb	up	gap (%)	t (s)
K-100-200-0.9-a-2	22.00	22.00	0.00	0.72
K-100-200-0.9-b-2	83.00	83.00	0.00	0.80
K-100-200-0.9-a-5	228.00	228.00	0.00	2.74
K-100-200-0.9-b-5	370.00	370.00	0.00	4.58
K-100-200-0.9-a-10	39.00	39.00	0.00	4.07
K-100-200-0.9-b-10	27.00	27.00	0.00	4.16
K-100-200-0.9-a-25	14.00	14.00	0.00	2.73
K-100-200-0.9-b-25	32.00	32.00	0.00	7.55
K-100-200-0.9-a-50	14.00	14.00	0.00	2.52
K-100-200-0.9-b-50	9.00	9.00	0.00	2.50
average			0.00	3.23

Table 6.1. Results for CPLEX Branch-and-bound Karařan instances with 100 nodes

instance	lb	up	gap (%)	t (s)
K-200-200-0.9-a-2	169,324.00	169,324.00	0.00	12,046.38
K-200-200-0.9-b-2	128,538.00	128,538.00	0.00	28,660.90
K-200-200-0.9-a-5	25,579.91	25,769.00	0.73	> 86,000.00
K-200-200-0.9-b-5	15,383.15	15,390.00	0.04	> 86,000.00
K-200-200-0.9-a-10	8,092.51	8,221.00	1.56	> 86,000.00
K-200-200-0.9-b-10	6,979.42	7,086.00	1.51	> 86,000.00
K-200-200-0.9-a-25	2,029.16	2,036.00	0.34	> 86,000.00
K-200-200-0.9-b-25	2,363.35	2,382.00	0.80	> 86,000.00
K-200-200-0.9-a-50	1,050.00	1,050.00	0.00	370.07
K-200-200-0.9-b-50	895.00	895.00	0.00	302.64
average			0.50	55,737.8

Table 6.2. Results for CPLEX Branch-and-bound for Karařan instances with 200 nodes

It can be seen that AU outperforms AM for both set of instances. For instances with up to 100 nodes, as reported in Table 6.3, AM finds a relative gap of 0.99%, while AU finds the optimal solution for all evaluated instances. As AMU ran AU, it also found the optimal solution for all evaluated instances. Both AU and AMU outperform CPLEX branch-and-bound in terms of running time, achieving an average running time of 0.02 seconds and 0.05 seconds respectively, while CPLEX branch-and-bound has an average running time of 3.23 seconds.

For instances with up to 200 nodes, as reported in Table 6.4, AM finds a relative

	CPLEX		AM		AU		AMU	
instance	gap (%)	t (s)	gap (%)	t (s)	gap (%)	t (s)	gap (%)	t (s)
K-100-200-0.9-a-2	0.00	0.72	0.00	0.01	0.00	0.01	0.00	0.01
K-100-200-0.9-b-2	0.00	0.80	0.00	0.01	0.00	0.02	0.00	0.02
K-100-200-0.9-a-5	0.00	2.74	0.00	0.02	0.00	0.01	0.00	0.03
K-100-200-0.9-b-5	0.00	4.58	0.26	0.01	0.00	0.01	0.00	0.03
K-100-200-0.9-a-10	0.00	4.07	0.00	0.02	0.00	0.01	0.00	0.06
K-100-200-0.9-b-10	0.00	4.16	0.00	0.02	0.00	0.02	0.00	0.05
K-100-200-0.9-a-25	0.00	2.73	0.00	0.04	0.00	0.04	0.00	0.10
K-100-200-0.9-b-25	0.00	7.55	3.03	0.03	0.00	0.04	0.00	0.09
K-100-200-0.9-a-50	0.00	2.52	6.67	0.05	0.00	0.05	0.00	0.10
K-100-200-0.9-b-50	0.00	2.50	0.00	0.04	0.00	0.04	0.00	0.09
average	0.00	3.23	0.99	0.02	0.00	0.02	0.00	0.05

Table 6.3. Results for AM, AU and AMU for Karařan instances with 100 nodes, compared with the CPLEX branch-and-bound

	CPLEX		AM		AU		AMU	
instance	gap (%)	t (s)	gap (%)	t (s)	gap (%)	t (s)	gap (%)	t (s)
K-200-200-0.9-a-2	0.00	12,046.38	0.01	0.02	0.01	0.02	0.01	0.07
K-200-200-0.9-b-2	0.00	28,660.90	0.69	0.02	0.69	0.02	0.69	0.06
K-200-200-0.9-a-5	0.73	> 86,000.00	2.65	0.05	2.61	0.06	2.61	0.13
K-200-200-0.9-b-5	0.04	> 86,000.00	2.08	0.02	2.08	0.02	2.08	0.05
K-200-200-0.9-a-10	1.56	> 86,000.00	9.53	0.10	9.90	0.02	9.53	0.24
K-200-200-0.9-b-10	1.51	> 86,000.00	6.74	0.10	4.17	0.12	4.17	0.24
K-200-200-0.9-a-25	0.34	> 86,000.00	3.05	0.21	2.21	0.20	2.21	0.50
K-200-200-0.9-b-25	0.80	> 86,000.00	4.13	0.22	3.94	0.22	3.94	0.48
K-200-200-0.9-a-50	0.00	370.07	2.86	0.31	2.23	0.34	2.23	0.72
K-200-200-0.9-b-50	0.00	302.64	4.48	0.34	4.58	0.38	4.48	0.78
average	0.50	55,737.80	3.62	0.13	3.24	0.14	3.19	0.33

Table 6.4. Results for AM, AU and AMU for Karařan instances with 200 nodes, compared with the CPLEX branch-and-bound

gap of 3.62%, while AU finds an relative gap of 3.24% and AMU achieves a relative gap of 3.19%. CPLEX branch-and-bound has a smaller gap of 0.50%, but an average running time of more than 55,000 seconds, while AM, AU and AMU running time never exceed 1 second.

As stated in Chapter 5, AM and AU consist in running two times the of Dijkstra's algorithm. Thus, it can justify the great time efficiency of these heuristics. Besides, as AM and AMU are 2-approximative algorithms for RSPT, they average case is closer to the optimum than it theoretical limit.

6.3 Scenario-based Algorithm

The third experiment aims to evaluate SBA for the proposed instances. First, it presents a sensibility analysis of SBA in relation to its parameters. Then, SBA is compared with AMU and CPLEX branch-and-bound.

In order to evaluate the SBA sensibility to its parameters, 12 different versions of SBA were developed and evaluated. Three different sets of initial and final scenarios are considered, being $\{0, 0.5\}$, $\{0.5, 1\}$, and $\{0, 1\}$. Moreover, four different step sizes are considered, being 0.1, 0.05, 0.01, and 0.001.

As AMU solves all instances with 100 vertices at optimality, this experiment is performed only in instances with 200 vertices. Table 6.5 shows the results of the sensibility analysis of SBA. The first column shows the step size γ . The second and third columns show the initial scenario α and the final scenario β , respectively. The fourth column reports the number of inspected scenario by SBA with the defined α, β , and γ . The fifth and sixth columns report, respectively, the average gap and the average running time over the proposed instances with 200 vertices for each combination of α, β , and γ reported in columns 1, 2 and 3.

γ	α	β	# scenarios	gap (%)	time (s)
0.1	0	0.5	6	2.67	0.97
	0.5	1	6	2.80	0.81
	0	1	11	2.48	1.55
0.05	0	0.5	11	2.59	1.55
	0.5	1	11	2.68	1.55
	0	1	21	2.41	2.98
0.01	0	0.5	51	1.87	7.25
	0.5	1	51	1.99	7.27
	0	1	101	1.73	14.28
0.001	0	0.5	501	1.75	69.86
	0.5	1	501	1.94	68.90
	0	1	1001	1.59	138.10

Table 6.5. Results for different versions of SBA

As shown in Table 6.5, SBA running time is directly correlated with the number of inspected scenarios, i.e. SBA's running time increases as the number of inspected scenarios increase. The works Coco et al. [2015, 2016] suggest that SBA finds good solutions when scenarios between the average scenario and the upper scenario are inspected, i.e. when $\alpha = 0.5$ and $\beta = 1$. However, one can see that, when SBA is

applied to solve RSPT, a smaller average gap can be achieved when inspecting solutions between the lower scenario and the median scenario than between the median scenario and the upper scenario, for all step sizes. However, some instances have a smaller gap between the median scenario and the upper scenario. Thus, the best average gap is achieved when SBA inspect solutions between the lower scenario and the upper scenario, for all step sizes. The best evaluated set of parameters for SBA is $\alpha = 0$, $\beta = 1$ and $\gamma = 0.001$, that achieved an average gap of only 1.59% in 138.10 seconds, in average.

Next, Table 6.6 reports the comparison of SBA with AMU and CPLEX branch-and-bound results presented in Tables 6.2 and 6.4. Two SBAs are analysed. The first, called SBA1, have $\alpha = 0$, $\beta = 1$ and $\gamma = 0.01$. The second, called SBA2, is the best evaluated SBA, as shown in Table 6.5. The first column reports the instances' name. The second and third columns report CPLEX branch-and-bound gap and running time. Then, each pair of columns refer to a heuristic. The first column of each pair shows the relative gap of the solution found by the heuristic over the lower bound of the CPLEX branch-and-bound, in percentage. The second column of each pair shows the heuristic running time in seconds.

Table 6.6. Results for CPLEX branch-and-bound, AMU and two different SBAs for Karařan instances with 200 vertices

instance	CPLEX		AMU		SBA1		SBA2	
	gap (%)	t (s)	gap (%)	t (s)	gap (%)	t (s)	gap (%)	t (s)
K-200-200-0.9-a-2	0.00	12,046.38	0.01	0.07	0.01	2.60	0.00	23.43
K-200-200-0.9-b-2	0.00	28,660.90	0.69	0.06	0.69	2.62	0.68	23.65
K-200-200-0.9-a-5	0.73	> 86,000.00	2.61	0.13	1.56	5.42	1.07	49.07
K-200-200-0.9-b-5	0.04	> 86,000.00	2.08	0.05	1.18	5.32	0.86	51.23
K-200-200-0.9-a-10	1.56	> 86,000.00	9.53	0.24	1.62	9.91	1.62	91.79
K-200-200-0.9-b-10	1.51	> 86,000.00	4.17	0.24	1.91	10.09	1.68	97.24
K-200-200-0.9-a-25	0.34	> 86,000.00	2.21	0.50	2.11	21.00	2.01	205.42
K-200-200-0.9-b-25	0.80	> 86,000.00	3.94	0.48	2.23	21.02	2.23	200.66
K-200-200-0.9-a-50	0.00	370.07	2.23	0.72	2.23	31.04	2.23	308.39
K-200-200-0.9-b-50	0.00	302.64	4.48	0.78	3.58	33.67	3.35	330.14
average	0.50	55,737.80	3.19	0.33	1.71	14.28	0.95	138.10

As can be seen in Table 6.6, SBA2 average gap is less than 1%, achieving a gap close to CPLEX branch-and-bound with an average running time two orders of magnitude faster. SBA1 improved 8 out of 10 instances when compared to AMU, and SBA2 improved 7 out of 10 instances when compared to SBA1. However, as shown in Table 6.5, a greater computational time is needed in order to lower the average gap. In general, as greater is the number of inspected scenarios, then lower is the average gap.

6.4 Mixed Integer Linear Programming Variable Neighbourhood Descent

This experiment aims to evaluate MILP-VND for the proposed instances. As AMU solves all instances with 100 vertices at optimality, this experiment is performed only in instances with 200 vertices. The results of this experiment are shown in Table 6.7. The first column reports the instances' name. The second and third columns report CPLEX branch-and-bound gap and running time. The fourth column shows the relative gap of the solution found by MILP-VND over the lower bound of the CPLEX branch-and-bound, in percentage. The last column reports the MILP-VND running time, in seconds.

Table 6.7. Results for CPLEX branch-and-bound and MILP-VND for Karaşan instances with 200 vertices

instance	CPLEX		MILP-VND	
	gap (%)	t (s)	gap (%)	t (s)
K-200-200-0.9-a-2	0.00	12,046.38	0.01	244.02
K-200-200-0.9-b-2	0.00	28,660.90	0.61	303.64
K-200-200-0.9-a-5	0.73	> 86,000.00	1.43	305.37
K-200-200-0.9-b-5	0.04	> 86,000.00	1.38	306.54
K-200-200-0.9-a-10	1.56	> 86,000.00	2.06	310.14
K-200-200-0.9-b-10	1.51	> 86,000.00	2.02	305.37
K-200-200-0.9-a-25	0.34	> 86,000.00	0.34	308.84
K-200-200-0.9-b-25	0.80	> 86,000.00	1.29	430.92
K-200-200-0.9-a-50	0.00	370.07	0.00	955.20
K-200-200-0.9-b-50	0.00	302.64	0.00	978.94
average	0.50	55,737.80	0.91	444.42

As can be seen in Table 6.7, MILP-VND can solve two instances at optimality, when CPLEX can solve five instances. It shows that the heuristic is commonly trapped into local optima, and can not converge to the global optimum in the most of the cases. However, MILP-VND average gap is 0.91%, and its average running time is 444.42 seconds. Thus, it presents a gap less than twice than CPLEX, and a running time two orders of magnitude faster.

6.5 Comparison of heuristics

The last experiment compares the heuristics AMU, SBA, and MILP-VND with CPLEX branch-and-bound based on the proposed MILP formulation (4.1)-(4.9). The results of these experiments is shown in Table 6.8. First column represents the instance name. Next, each pair of consecutive columns refer to one heuristic, AMU, SBA, and MILP-VND, respectively. The first column of each pair shows the relative gap of the solution found by the heuristic over the lower bound of the CPLEX branch-and-bound, in percentage. The second column of each pair shows the algorithm running time in seconds. As instances with 100 vertices are easily solved by CPLEX or AMU, Table 6.8 shows results only for instances with 200 vertices.

Table 6.8. Results for CPLEX Branch-and-bound and heuristics for Karaşan instances with 200 nodes

instance	AMU		SBA		MILP-VND	
	gap (%)	t (s)	gap (%)	t (s)	gap (%)	t (s)
K-200-200-0.9-a-2	0.01	0.07	0.00	23.43	0.01	244.02
K-200-200-0.9-b-2	0.69	0.06	0.68	23.65	0.61	303.64
K-200-200-0.9-a-5	2.61	0.13	1.07	49.07	1.43	305.37
K-200-200-0.9-b-5	2.08	0.05	0.86	51.23	1.38	306.54
K-200-200-0.9-a-10	9.53	0.24	1.62	91.79	2.06	310.14
K-200-200-0.9-b-10	4.17	0.24	1.68	97.94	2.02	305.37
K-200-200-0.9-a-25	2.21	0.50	2.01	205.42	0.34	308.84
K-200-200-0.9-b-25	3.94	0.48	2.23	200.66	1.29	430.92
K-200-200-0.9-a-50	2.23	0.72	2.23	308.39	0.00	955.20
K-200-200-0.9-b-50	4.48	0.78	3.35	330.14	0.00	978.94
average	3.19	0.33	0.95	138.10	0.91	444.42

It can be seen in Table 6.8 that the fastest algorithm is AMU. It achieves a gap of 3.19% with a running time that never exceeds 1 second. However, MILP-VND produces the best gap of all heuristics. It gives a gap of 0.91%, just 0.41% greater than CPLEX branch-and-bound. However, its computational cost was more than 3 times higher than SBA.

Chapter 7

Conclusions

In this dissertation, the Robust Shortest Path Tree (RSPT) problem with interval data and minmax regret criteria was proposed. It was inspired by a real network application, the RPL routing problem. The RPL protocol builds a routing table for 6LoWPANs, that is a special type of network that composes the Internet of Things. This problem consists in computing a spanning tree in a given graph, such that the sum of the robust cost of the paths between the root node s and every other node of the network is minimized. In this work, the RSPT problem was formally defined, and a mathematical formulation was proposed. Besides, a set of algorithms for RSPT were presented and evaluated.

As far as we know, this is the first work in literature that develops optimization algorithms for the RPL protocol. Another novel contribution of this work is to use Robust Optimization techniques in order to consider the channel variability of 6LoWPANs. Thus, this work proposes to extend the RPL protocol with the AU heuristic as objective function. As AU simple consists in running the Dijkstra's algorithm in a pre-defined scenario, it can be implemented as a fully distributed network protocol.

To our knowledge, this is the first work in literature that deals with the RSPT. In order to compare the developed algorithms, a heuristic and three approximative algorithms for others Robust Optimization problems presented in [Kasperski and Zieliński, 2006] and [Coco et al., 2015] were extended for the RSPT. A proof that three of these algorithm holds a 2-approximation factor for the RSPT was developed. Besides, a Mixed Integer Linear Programming based local search was proposed. MILP-VND average gap was just 0.41% greater than CPLEX branch-and-bound, but its running time was 2 orders of magnitude lower. Thus, it was considered the best heuristic to solve RSPT.

Future works may use the developed model to compute the DODAG used by

the RPL for routing packets in 6LoWPANs, using the AU heuristic to choose the arcs that will be at the routing table. Furthermore, it is proposed to improve the presented formulations for the RSPT, and proposes new mathematical formulations for this problem, so as the development of new exact and heuristics algorithms for the RSPT. At last, it is proposed to extend the developed MILP-VND heuristic to other robust optimization problems that AMU or SBA can be efficiently used.

Bibliography

- Aissi, H., Bazgan, C., and Vanderpooten, D. (2009). Min-max and min-max regret versions of combinatorial optimization problems: A survey. *European Journal of Operational Research*, 197(2):427--438.
- Aron, I. and Hentenryck, P. V. (2002). A constraint satisfaction approach to the robust spanning tree problem with interval data. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, pages 18--25. Morgan Kaufmann Publishers Inc.
- Aron, I. D. and Hentenryck, P. V. (2004). On the complexity of the robust spanning tree problem with interval data. *Operations Research Letters*, 32(1):36--40.
- Assunção, L., de Noronha, T. F., Santos, A. C., and de Andrade, R. C. (2014). A linear programming based heuristic for robust optimization problems: a case study on solving the restricted robust shortest path problem. In *Matheuristics 2014-5th International Workshop on Model-Based Metaheuristics, Hambourg, Alemagne*, page 8p.
- Atzori, L., Iera, A., and Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, 54(15):2787--2805.
- Averbakh, I. (2001). On the complexity of a class of combinatorial optimization problems with uncertainty. *Mathematical Programming*, 90(2):263--272.
- Averbakh, I. (2005). Computing and minimizing the relative regret in combinatorial optimization with interval data. *Discrete Optimization*, 2:273--287.
- Bellman, R. (1956). On a routing problem. Technical report, DTIC Document.
- Ben-Tal, A. and Nemirovski, A. (2002). Robust optimization--methodology and applications. *Mathematical Programming*, 92(3):453--480.
- Bondy, J. A. and Murty, U. S. R. (1976). *Graph theory with applications*, volume 290. Macmillan London.

- Candia-Véjar, A., Álvarez-Miranda, E., and Maculan, N. (2011). Minmax regret combinatorial optimization problems: an algorithmic perspective. *RAIRO-Operation Research*, 45:101--129.
- Catanzaro, D., Labbé, M., and Salazar-Neumann, M. (2011). Reduction approaches for robust shortest path problems. *Computers & Operations Research*, 38:1610--1619.
- Chakeres, I. and Perkins, C. (2008). Dynamic manet on-demand (dymo) routing. Technical report, Internet Engineering Task Force.
- Chang, J. M., Yang, H. Y., Chao, H. C., and Chen, J.-L. (2010). Multipath design for 6lowpan ad hoc on-demand distance vector routing. *International Journal of Information Technology, Communications and Convergence*, 1(1):24--40.
- Chekuri, C., Shepherd, F. B., Oriolo, G., and Scutellà, M. G. (2007). Hardness of robust network design. *Networks*, 50(1):50--54.
- Coco, A. A., Júnior, J. C. A., Noronha, T. F., and Santos, A. C. (2014a). An integer linear programming formulation and heuristics for the minmax relative regret robust shortest path problem. *Journal of Global Optimization*, 60(2):265--287. ISSN 0925-5001.
- Coco, A. A., Santos, A. C., and Noronha, T. F. (2015). Senario-based heuristics with path-relinking for the robust set covering problem. In *Proceedings of the XI Metaheuristics International Conference (MIC)*.
- Coco, A. A., Santos, A. C., and Noronha, T. F. (2016). Coupling scenario-based heuristics to exact methods for the robust weighted set covering problem with interval data. In *VIII Conference on Manufacturing, Modelling, Management & Control*.
- Coco, A. A., Solano-Charris, E. L., Santos, A. C., Prins, C., and Noronha, T. F. (2014b). Robust optimization criteria: state-of-the-art and new issues. Technical report UTT-LOSI-14001, Université de Technologie de Troyes.
- Conde, E. (2012). On a constant factor approximation for minmax regret problems using a symmetry point scenario. *European Journal of Operational Research*, 219:452--457.
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). Single-source shortest paths. In *Introduction to Algorithms*, pages 580--619. MIT Press Cambridge.

- Daniels, R. L. and Kouvelis, P. (1995). Robust scheduling to hedge against processing time uncertainty in single-stage production. *Management Science*, 41(2):363--376.
- Degel, D. and Lutter, P. (2013). A robust formulation of the uncertain set covering problem. *Working paper, Bochum*.
- Dei, V. G., Woeginger, G. J., et al. (2006). On the robust assignment problem under a fixed number of cost scenarios. *Operations Research Letters*, 34(2):175--179.
- Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269--271.
- Ee, G. K., Ng, C. K., Noordin, N. K., and Ali, B. M. (2010). A review of 6lowpan routing protocols. *Proceedings of the Asia-Pacific Advanced Network*, 30:71--81.
- Felsche, M., Huhn, A., and Schwetlick, H. (2012). Routing protocols for 6lowpan. In *IT Revolutions*, pages 71--83. Springer.
- Furdík, K. and Lukác, G. (2012). Events processing and device interoperability in a smart office iot application. In *Proceedings of the 23rd Central European Conference on Information and Intelligent Systems (CECIIS 2012)*, University of Zagreb, Croatia, pages 387--394.
- Gabrel, V., Murat, C., and Thièle, A. (2013). La pw-robustesse: pourquoi un nouveau critère de robustesse et comment l'appliquer? In *14eme congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF)*.
- Gaddour, O. and Koubâa, A. (2013). Rpl in a nutshell: A survey. *Computer Networks*, 56(14):3163--3178.
- Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-completeness*. WH Freeman and Company, New York.
- Ghaoui, L. E. and Lebret, H. (1997). Robust solutions to least-squares problems with uncertain data. *SIAM Journal on Matrix Analysis and Applications*, 18(4):1035--1064.
- Giusto, D., Lera, A., Morabito, G., and Atzori, L. (2010). *The Internet of Things*. Springer.
- Gubbi, J., Buyya, R., Marusic, S., and Palaniswami, M. (2013). Internet of things (iot): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645--1660.

- Gupta, S. K. and Rosenhead, J. (1968). Robustness in sequential investment decisions. *Management science*, 15(2):B--18.
- Gutiérrez, G. J., Kouvelis, P., and Kurawarwala, A. A. (1996). A robustness approach to uncapacitated network design problems. *European Journal of Operational Research*, 94(2):362--376.
- Hansen, P. and Mladenović, N. (2001). Variable neighborhood search: Principles and applications. *European journal of operational research*, 130(3):449--467.
- Kalaš, R., Lamboray, C., and Vanderpooten, D. (2012). Lexicographic α -robustness: An alternative to min-max criteria. *European Journal of Operational Research*, 220(3):722--728.
- Karaşan, O. E., Yaman, H., and Pinar, M. C. (2001). The robust shortest path problem with interval data. Technical report, Bilkent University, Department of Industrial Engineering.
- Karp, R. M. (1972). Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85--103. Springer.
- Kasperski, A., Makuchowski, M., and Zieliński, P. (2012). A tabu search algorithm for the minmax regret minimum spanning tree problem with interval data. *Journal of Heuristics*, 18(4):593--625.
- Kasperski, A. and Zieliński, P. (2006). An approximation algorithm for interval data minmax regret combinatorial optimization problems. *Information Processing Letters*, 97(5):177--180.
- Kasperski, A. and Zieliński, P. (2007). On the existence of an fptas for minmax regret combinatorial optimization problems with interval data. *Operations Research Letters*, 35(4):525--532.
- Kim, K., Montenegro, G., Park, S., Chakeres, I., and Perkins, C. (2007a). Dynamic manet on-demand for 6lowpan (dymo-low) routing. Technical report, Internet Engineering Task Force.
- Kim, K., Park, S. D., Montenegro, G., Yoo, S., and Kushalnagar, N. (2007b). 6lowpan ad hoc on-demand distance vector routing (load). Technical report, Internet Engineering Task Force.

- Kim, K., Yoo, S., Park, J., Park, S. D., and Lee, J. (2005). Hierarchical routing over 6lowpan (hilow). Technical report, Internet Engineering Task Force.
- Koksal, C. E., Jamieson, K., Telatar, E., and Thiran, P. (2006). Impacts of channel variability on link-level throughput in wireless networks. In *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems, SIGMETRICS '06/Performance '06*, pages 51--62, New York, NY, USA. ACM.
- Kortuem, G., Kawsar, F., Fitton, D., and Sundramoorthy, V. (2010). Smart objects as building blocks for the internet of things. *Internet Computing, IEEE*, 14(1):44--51.
- Kouvelis, P. and Yu, G. (1997). *Robust discrete optimization and its applications*, volume 14 of *Nonconvex optimization and its applications*. Springer.
- Kushalnagar, N., Montenegro, G., and Schumacher, C. (2007). Ipv6 over low-power wireless personal area networks (6lowpans): overview, assumptions, problem statement, and goals. Technical report, Internet Engineering Task Force.
- Lee, C. C., Lee, W. C., Cai, H., Chi, H. R., Wu, C. K., Haase, J., and Gidlund, M. (2015). Traffic condition monitoring using weighted kernel density for intelligent transportation. In *Industrial Informatics (INDIN), 2015 IEEE 13th International Conference on*, pages 624--627. IEEE.
- Lin, S. (1965). Computer solutions of the traveling salesman problem. *Bell System Technical Journal, The*, 44(10):2245--2269.
- Madakam, S., Ramaswamy, R., and Tripathi, S. (2015). Internet of things (iot): A literature review. *Journal of Computer and Communications*, 3(05):164.
- Miorandi, D., Sicari, S., De Pellegrini, F., and Chlamtac, I. (2012). Internet of things: Vision, applications and research challenges. *Ad Hoc Networks*, 10(7):1497--1516.
- Monaci, M. and Pferschy, U. (2013). On the robust knapsack problem. *SIAM Journal on Optimization*, 23(4):1956--1982.
- Montemanni, R. (2006). A benders decomposition approach for the robust spanning tree problem with interval data. *European Journal of Operational Research*, 174(3):1479--1490.
- Montemanni, R., Barta, J., and Gambardella, L. (2006). Heuristic and preprocessing techniques for the robust traveling salesman problem with interval data. Technical report Technical report IDSIA-01-06, Istituto Dalle Molle di Studi sull'Intelligenza Artificiale.

- Montemanni, R., Barta, J., Mastrolilli, M., and Gambardella, L. M. (2007). The robust traveling salesman problem with interval data. *Transportation Science*, 41(3):366--381.
- Montemanni, R. and Gambardella, L. M. (2005a). A branch and bound algorithm for the robust spanning tree problem with interval data. *European Journal of Operational Research*, 161(3):771--779.
- Montemanni, R. and Gambardella, L. M. (2005b). The robust shortest path problem with interval data via Benders decomposition. *4OR*, 3(4):315--328.
- Montemanni, R., Gambardella, L. M., and Donati, A. V. (2004). A branch and bound algorithm for the robust shortest path problem with interval data. *Operations Research Letters*, 32(3):225--232.
- Pereira, J. and Averbakh, I. (2011). Exact and heuristic algorithms for the interval data robust assignment problem. *Computers & Operations Research*, 38(8):1153--1163.
- Pérez, F., Astudillo, C. A., Bardeen, M., and Candia-Véjar, A. (2012). A simulated annealing approach for the minmax regret path problem. In *Proceedings of the Congreso Latino Americano de Investigación Operativa (CLAIO) - Simpósio Brasileiro de Pesquisa Operacional (SBPO)*.
- Perkins, C., Belding-Royer, E., and Das, S. (2003). Ad hoc on-demand distance vector (aodv) routing. Technical report, Internet Engineering Task Force.
- Roy, B. (2010). Robustness in operational research and decision aiding: A multi-faceted issue. *European Journal of Operational Research*, 200(3):629--638.
- Royer, E. M. and Perkins, C. E. (1999). Multicast operation of the ad-hoc on-demand distance vector routing protocol. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 207--218. ACM.
- Said, O. and Masud, M. (2013). Towards internet of things: Survey and future vision. *International Journal of Computer Networks*, 5(1):1--17.
- Shelby, Z. and Bormann, C. (2011). *6LoWPAN: The wireless embedded Internet*, volume 43 of *Wiley Series in Communications Networking & Distributed Systems*. John Wiley & Sons.
- Spall, J. C. (2005). *Introduction to stochastic search and optimization: estimation, simulation, and control*, volume 65 of *Wiley Series in Discrete Mathematics and Optimization*. John Wiley & Sons.

- Sugiyama, K., Tagawa, S., and Toda, M. (1981). Methods for visual understanding of hierarchical system structures. *Systems, Man and Cybernetics, IEEE Transactions on*, 11(2):109--125.
- Talbi, E.-G. (2009). *Metaheuristics: from design to implementation*, volume 74. John Wiley & Sons.
- TongKe, F. (2013). Smart agriculture based on cloud computing and iot. *Journal of Convergence Information Technology*, 8(2).
- Ukkusuri, S. V., Mathew, T. V., and Waller, S. T. (2007). Robust transportation network design under demand uncertainty. *Computer-Aided Civil and Infrastructure Engineering*, 22(1):6--18.
- Vasseur, J., Agarwal, N., Hui, J., Shelby, Z., Bertrand, P., and Chauvenet, C. (2011). RPL: The IP routing protocol designed for low power and lossy networks. *Internet Protocol for Smart Objects (IPSO) Alliance*.
- Winter, T. (2012). RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. Technical report, Internet Engineering Task Force.
- Wu, B. Y. and Chao, K. (2004). Shortest-paths trees. In *Spanning Trees and Optimization Problems*, pages 23--40. CRC Press.
- Wu, C. W., Brown, K. N., and Beck, J. C. (2009). Scheduling with uncertain durations: Modeling β -robust scheduling with constraints. *Computers & Operations Research*, 36(8):2348--2356.
- Xia, F., Yang, L. T., Wang, L., and Vinel, A. (2012). Internet of things. *International Journal of Communication Systems*, 25(9):1101.
- Yaman, H., Karas¸an, O. E., and P¸inar, M. ¸. (2001). The robust spanning tree problem with interval data. *Operations Research Letters*, 29(1):31--40.
- Yu, G. (1996). On the max-min 0-1 knapsack problem with robust optimization applications. *Operations Research*, 44(2):407--415.
- Zanella, A., Bui, N., Castellani, A., Vangelista, L., and Zorzi, M. (2014). Internet of things for smart cities. *Internet of Things Journal, IEEE*, 1(1):22--32.
- Zhu, C., Zheng, J., Ngo, C., Park, T., Zhang, R., and Lee, M. (2009). Low-rate wpan mesh network-an enabling technology for ubiquitous networks. In *Wireless*

Communications and Networking Conference, 2009. WCNC 2009. IEEE, pages 1--6.
IEEE.