PROBLEMA DA DEPOSIÇÃO GAMMA EM GRADES: COMPLEXIDADE E NOVA FORMULAÇÃO DE PROGRAMAÇÃO LINEAR INTEIRA

MARCELO FONSECA FARAJ

PROBLEMA DA DEPOSIÇÃO GAMMA EM GRADES: COMPLEXIDADE E NOVA FORMULAÇÃO DE PROGRAMAÇÃO LINEAR INTEIRA

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: Sebastián Alberto Urrutia
Coorientador: João Fernando Machry Sarubbi

Belo Horizonte
Fevereiro de 2019

MARCELO FONSECA FARAJ

GAMMA DEPLOYMENT PROBLEM IN GRIDS: COMPLEXITY AND A NEW INTEGER LINEAR PROGRAMMING FORMULATION

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 : Sebastián Alberto Urrutia
Co-Advisor : João Fernando Machry Sarubbi

Belo Horizonte
February 2019

Ficha catalográfica elaborada pela Biblioteca do ICEx - UFMG

Faraj, Marcelo Fonseca

F219 g Gamma deployment problem in grids: complexity and a new integer linear programming formulation / Marcelo Fonseca Faraj — Belo Horizonte, 2019.
xxiv, 44 p.: il.; 29 cm.

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

Orientador: Sebastián Alberto Urrutia
Coorientador: João Fernando Machry Sarubbi

1. Computação – Teses. 2. Problema da deposição gamma. 3. Redes veiculares. 4. Complexidade computacional. 5. Programação linear inteira. 6. Heurística. 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

Gamma Deployment Problem in Grids: Complexity and a new Integer Linear Programming Formulation

MARCELO FONSECA FARAJ

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. SEBASTIÁN ALBERTO URRUTIA - Orientador
Departamento de Ciência da Computação - UFMG

PROF. JOÃO FERNANDO MACHRY SARUBBI - Coorientador
Departamento de Computação - CEFET-MG

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

PROF. CRISTIANO MACIEL DA SILVA
Departamento de Tecnologia - UFSJ

Belo Horizonte, 15 de fevereiro de 2019.

I dedicate this work to the myth-poetic voice, whose death means the emptying of science

Acknowledgments

I should start by thanking Dr. Sebastián Urrutia and Dr. João Sarubbi for their important role in my Master's. This work would not have been possible without their valuable advices.

I thank my mother and father for being the greatest parents in the world and for always lighting me with their wisdom and support. I thank Yara, an amazing woman I was lucky to know, date, and marry during the period of my Master's. I thank my five siblings, their spouses, and children for playing an essential role in my beautiful and united family. I thank my grandparents, uncles, aunts, cousins, friends, colleagues, and professors. Even my haters deserve my thanks — they challenge my comfort zone.

Last in order but first in importance, I thank God for my life, conscience, and virtues.

"Assim como a leitura, a mera experiência não pode substituir o pensamento. A pura empiria está para o pensamento como o ato de comer está para a digestão e a assimilação."
(Arthur Schopenhauer)

Abstract

Vehicular networks are one of the most significant components of intelligent transportation systems. They have the potential to ease traffic management, lower accident rates and provide other solutions to smart cities. One of the main challenges on vehicular networks is to choose the best places to deploy roadside units. This thesis deals with the Gamma Deployment Problem, which consists in deploying the minimum number of roadside units on a road network meeting the Gamma Deployment metric. Within this metric, at least a given fraction of vehicles passing in the road network must be covered, i.e they should meet at least one roadside unit each predetermined time interval. In this thesis, we propose a formal treatment based on graph theoretical concepts and provide a proof that the decision version of the Gamma Deployment Problem in Grids is NP-complete. In addition, we expose an issue in the multi-flow integer linear programming formulation present in literature and propose a slight correction for it. We also introduce a new integer linear programming formulation based on set covering and provide a proof that the polytope associated with its linear programming relaxation is contained in the polytope associated with the linear programming relaxation of the multi-flow formulation. Finally, computational experiments with a commercial optimizer show that the set covering formulation widely outperforms the multi-flow formulation regarding linear programming relaxation gap and execution time.

Keywords: Gamma Deployment Problem, Vehicular Networks, Complexity, Integer Linear Programming, Heuristic.

Resumo

Redes veiculares constituem um dos componentes mais importantes dos sistemas inteligentes de transporte. Elas possuem potencial para facilitar a gestão de tráfego, reduzir taxas de acidente de trânsito e proporcionar outras soluções para a construção de cidades inteligentes. Um dos principais desafios associados a redes veiculares é a escolha das melhores localizações para instalação das infraestruturas de comunicação, as quais são conhecidas como *roadside units*. Esta dissertação lida com o Problema da Deposição Gamma, o qual consiste em instalar o menor número possível de unidades de infraestrutura em uma rede rodoviária de modo a cumprir a métrica Deposição Gamma. De acordo com esta métrica, uma porcentagem mínima dos veículos transitando pela rede rodoviária devem estar cobertos, sendo que um veículo é considerado coberto caso encontre ao menos uma unidades de infraestrutura em cada intervalo de tempo de duração pré-determinada durante a sua viagem. Nesta dissertação, introduz-se um tratamento baseado em teoria dos grafos para o Problema da Deposição Gamma e apresenta-se uma prova de que a versão de decisão do Problema da Deposição Gamma em Grades pertence à classe de complexidade NP-completo. Em seguida, expõe-se uma imperfeição no modelo multifluxo de programação linear inteira presente na literatura e propõe-se uma pequena correção. Também se introduz um novo modelo de programação linear inteira baseado em cobertura de conjuntos e demonstra-se que o politopo associado a sua relaxação linear está contido no politopo associado à relaxação linear do modelo multifluxo. Por fim, experimentos computacionais com um otimizador comercial mostram que a formulação de cobertura de conjuntos se comporta de modo significativamente superior à formulação multifluxo em termos de gap de relaxação linear e tempo de execução.

Palavras-chave: Problema da Deposição Gamma, Redes Veiculares, Complexidade, Programação Linear Inteira, Heurística.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

The number of cars and other vehicles used on a daily basis is growing and one of the most significant issues related to it is the increasing rate of fatalities due to accidents on roads [14, 22]. Vehicular networks are considered a mobile ad hoc network (MANETs) aiming to address this situation and provide other benefits to drivers and society. Specifically designed for the domain of vehicles, roads, and pedestrians, vehicular networks are considered one of the essential technologies to enable intelligent transportation systems, which are aligned with the concept of smart city.

According to Eze et al. [7], vehicular network applications are typically classified either as safety–related or as non-safety–related. As safety applications, they list three categories: (i) driver assistance, comprising cooperative collision avoidance, road navigation, and lane changing; (ii) alert information, including work zone and speed limit; and (iii) warning alert, comprising warnings about road obstacles, post-crash, and other life-threatening traffic conditions. As non-safety or commercial applications, they include those aiming to: (i) improve traffic efficiency; (ii) guarantee passengers comfort; and (iii) provide commercial advertisements.

Kumar et al. [14] propose a different classification in which vehicular network applications are categorized into four classes: (i) safety oriented, which includes real-time traffic, post-crash notification, cooperative collision warning, and traffic vigilance against driving offenses; (ii) commerce oriented, which includes Internet access, digital map downloading, and value-added advertisement; (iii) convenience oriented, which includes parking availability, active prediction of upcoming topography to optimize fuel, electronic toll collection, and route diversions to avoid traffic jam; and (iv) productivity oriented, which covers environmental benefits, fuel saving, and productive time utilization during road congestions.

Vehicular networks have three principal physical components [22]: (i) fixed in-

frastructure units called roadside units (RSUs); (ii) vehicles with communicative capabilities; and (iii) a communication channel with dynamic capacity. From a structural perspective, RSUs and vehicles are usually called nodes of the vehicular network.

Direct communication between nodes of a vehicular network can be classified into three classes: (i) inter-vehicle, also called vehicle-to-vehicle (V2V); (ii) inter-roadside, also called infrastructure-to-infrastructure (I2I); and (iii) vehicle-to-roadside, also called vehicle-to-infrastructure (V2I). Figure 1.1 presents these three communication classes, which can co-occur in practice. I2I communication depends typically on a wired network interconnecting RSUs. V2V communication relies on wireless messages exchanged between vehicles, and V2I communication relies on wireless messages exchanged between RSUs and vehicles. Additionally, vehicular networks allow multi-hop communication, which means that one or more nodes can work as intermediary in the transmission of messages between two other nodes in the network.

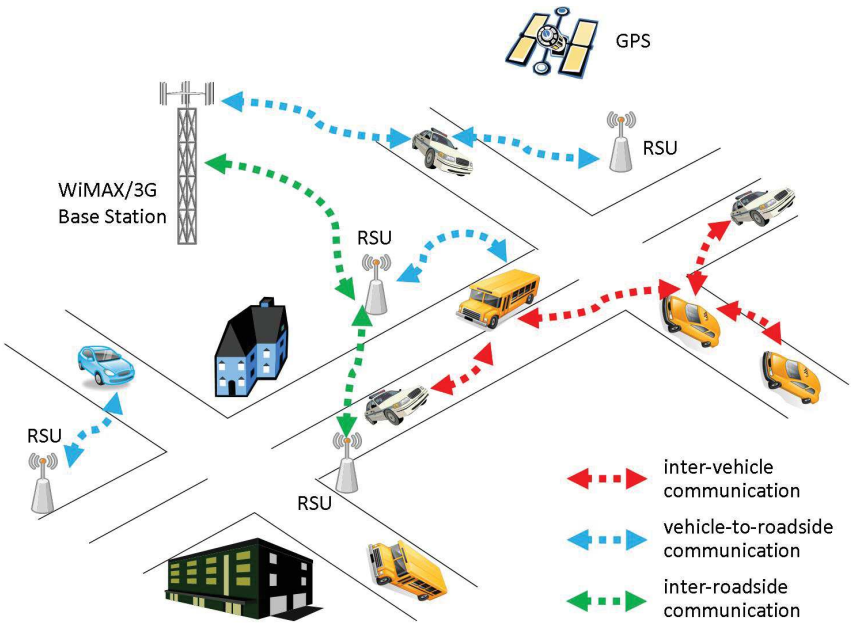


Figure 1.1: The three different types of communication present in vehicular networks [6].

Saini et al. [22] compare vehicular networks and general MANETs regarding many aspects. They share some characteristics, such as the capability of improving coverage through multi-hop communication. On the other hand, there are many differences between them, which causes typical routing protocols and metrics of MANETs to be inconsistent for the domain of vehicular networks [5, 7]. Some examples of these differences follow:

- MANETs have only mobile nodes, while vehicular networks have mobile and static nodes;

- MANETs use no infrastructure, while vehicular networks can use RSUs as gateways to the Internet;

- MANETs have energy constraints related to mobile nodes battery, which vehicular networks do not;

- MANETs have unconstrained mobility patterns, while vehicular networks have quick movements constrained by the available roads;

- MANETs usually provide longer connection life to its nodes, while vehicular networks usually provide short connection life depending on traffic jams, road conditions, and other factors.

Since vehicular networks are a relatively recent technology, many of their aspects still demand research, standardization, and development, which explains current attention from car manufacturing industries, academia, and government agencies [7]. Some of the most common research areas are routing, protocols, broadcasting, metrics, security, and — our focus in this work — the deployment of RSUs on a road network.

Many works have shown that V2I communication provides a significant improvement in the performance of a vehicular network [17, 21, 34]. Nevertheless, a central constraint associated with this kind of communication is the high cost of installing and maintaining RSUs [3, 13]. Hence, a big challenge related to the project of a vehicular network consists in minimizing the number of installed RSUs while still guaranteeing some quality of service (QoS) metric. By QoS metric, we mean a formal guarantee related to the availability of RSUs in the road network.

In this work, we specifically deal with a problem we name Gamma Deployment Problem (GDP): deploying the minimum number of RSUs in a road network while ensuring a QoS metric called Gamma Deployment [25, 26]. This metric provides two parameters to be set by the designer of a vehicular network: (i) a minimum V2I inter-contact time for covered vehicles; and (ii) a minimum fraction of vehicles to be covered. Throughout this work, we focus on road networks discretized into grid graphs (for a formal definition of grid graphs, refer to [12]). For this reason, most of our contributions are directed at the Gamma Deployment Problem in Grids (GDPG).

1.1 Related Works

The problem of deploying RSUs to compose a vehicular network has been studied by many authors and with a wide variety of QoS metrics, constraints, and objective functions. In this section, we briefly list some of those works to give an idea about the context in which our work is inserted.

Lochert et al. [16] implemented a genetic algorithm which tries to find the optimal deployment of a limited number of RSUs for a vehicular network traffic information system while maximizing the travel savings for some fixed landmarks.

Sun and Yang [29] proposed a significance ranking model and three strategies for computing significance degrees, which led to a heuristic for deploying RSUs. They evaluated the performance of the deployment using a simulation tool.

Trullols-Cruces et al. [31] developed an integer quadratic programming model to deploy a limited number of RSUs aiming to provide Internet access services for the maximum road traffic volumes.

Xiong et al. [33] tackled the minimal deployment of RSUs based on the vehicular mobility pattern. They proposed a graph model to characterize the observed pattern and a greedy heuristic for deploying the RSUs.

Liu et al. [15] proposed a file downloading oriented strategy for deploying RSUs. Their strategy involved modeling encounters between RSUs and vehicles as time continuous Markov chains to find the optimal inter-meeting time.

Aslam et al. [2] presented a binary linear programming formulation and a balloon expansion heuristic for deploying a limited number of RSUs in an urban region to maximize the information flow from vehicles to RSUs. Besides the differences between their work and ours related to the constraints and objective function, they assume that roads likely to have low traffic can be removed from the algorithm execution, which we do not.

Trullols et al. [30] formulated the deployment of a limited number of RSUs as a maximum coverage problem in which the deployment aimed at maximizing the number of vehicles contacting RSUs and the V2I connection time. They also dealt with another version of the problem in which each covered vehicle should contact RSUs for at least a given time interval. They formulated this version as a maximum coverage problem with time threshold. They proposed heuristics for both problems.

Silva and Meira [28] proposed a QoS metric called Delta Deployment, which ensures that at least a fraction of the vehicles in the road network are covered during a given percentage of their trips. They also proposed a greedy heuristic to minimize the number of RSUs deployed to ensure this metric. Other works have also proposed

heuristics to deploy RSUs based on the Delta Deployment metric. For instance, Sarubbi and Silva [24] proposed another greedy heuristic, and Sarubbi et al. [23] proposed a genetic algorithm. The difference between Delta Deployment and Gamma Deployment makes the problem tackled by these works incompatible with a metric based on the percentage of time a vehicle trip is covered. Gamma Deployment, instead, is a metric based on the inter-contact time of vehicles.

Rashidi et al. [20] analyzed the trade-offs between the length of gaps between RSUs and other parameters related to the quality of communication, such as data delivery ratio. They also introduced some theoretical upper bounds and a heuristic to determine the distance between neighboring RSUs. Their approach and Gamma Deployment aim at the same vehicular network applications: delay-tolerant applications. Nevertheless, they tackle the deployment from a perspective of spatial distance — which allows them to abstractly express their results in terms of distance between RSUs — while Gamma Deployment is concerned about the inter-contact time between vehicles and RSUs.

Zheng et al. [35] introduced the idea of intermittent coverage for mobile nodes. They proposed a QoS metric called α-Coverage, whose concept is based on providing worst-case guarantees on V2I inter-contact distance for every possible path in a road network. A deployment of RSUs in a road network complies with this metric if at least one RSU is deployed at each possible path whose length is larger than or equal to α. They also proved that verifying if a deployment provides α-Coverage is NP-complete and deploying a limited number of RSUs to provide α-Coverage is NP-hard.

The work of Zheng et al. [35] is important to contextualize our work since the Gamma Deployment metric fits in the notion of intermittent coverage. Nevertheless, Gamma Deployment differs from α-Coverage in at least two essential aspects, which causes the computational complexity bounds proved in [35] not to be extensible for the problem of deploying a limited number of RSUs to meet a Gamma Deployment. These two aspects are: (i) α-Coverage provides inter-contact distance guarantees, while Gamma Deployment provides inter-contact time guarantees; (ii) α-Coverage guarantees hold for any possible path in the road network, while Gamma Deployment guarantees only hold for a given set of vehicle trips in the road network — Gamma Deployment could not behave differently since two vehicles can travel a path with different speeds.

Silva et al. [25] proposed Gamma Deployment, the QoS metric in which our work is based. They also introduced a greedy heuristic called Gamma-g to solve what we call GDPG and showed that it deployed considerably fewer RSUs than the intuitive method of placing RSUs at the densest locations. Next, Silva et al. [26] proposed an integer linear programming (ILP) formulation based on multi-flow to solve GDPG and

compared its results with those from Gamma-g for small instances. We detail this formulation at Section 4.1. Then, Faraj et al. [8, 9] proposed a memetic algorithm called Gamma-LSGA to solve GDPG, which highly outperforms Gamma-g.

Last, we mention that this work also enters the theoretical context of a rich class of problems known as Facility Location Problems. This class include wide subclasses of problems such as Median Problems, Covering Problems, Center Problems, and so forth. For some reviews on Facility Location Problems, refer to [18], [19].

1.2 Motivation and Contributions

In this section, we list the main motivations and contributions of this work.

Our first motivation is the fact that here is no approach in the literature dealing with GDP under the tools of graph theory. This manuscript introduces this attitude by formalizing a definition of GDP with graph theoretical concepts (Section 2.2) and by tackling the problem from this perspective throughout the text.

So far, GDPG has only been approached from a practical perspective, with heuristics and an integer linear programming formulation to solve it. Hence, our second motivation is the lack of a theoretical research on its computational complexity. In this work, we provide a proof that the decision version of GDPG is NP-complete.

Despite working properly for many instances, the multi-flow ILP formulation proposed by Silva et al. [26] is incorrect when there is at least one cycle in one or more of the trips in an instance. This is our third motivation and, to overcome this situation, we propose a slight correction that does not affect the main idea of the formulation but entirely fixes the problem.

Our last motivation is an intent of achieving better running times and being able to run more instances than the multi-flow ILP model. In this work, we propose a new ILP formulation based on set covering with considerably fewer variables and constraints and whose linear programming relaxation polytope is contained in the corresponding polytope of the multi-flow formulation. We experimentally show that the set covering model outperforms the multi-flow model regarding running time and linear programming relaxation bound.

1.3 Outline

The remainder of this dissertation is organized as follows: in Chapter 2, we explain the Gamma Deployment metric, formally define GDP with graph theory, and explain

the discretization method used to abstractly express any road network as a grid (to convert any GDP instance into a GDPG instance); in Chapter 3, we present a detailed proof that the decision version of GDPG is NP-complete; in Chapter 4, we present the multi-flow formulation to solve GDPG, show and correct an issue with it, and propose a new set covering formulation with the proof that its polytope is contained in the polytope of the multi-flow formulation; in Chapter 5, we describe and analyze some computational experiments to compare the corrected multi-flow formulation and the set covering formulation; in Chapter 6, we conclude this work with final remarks and suggestions for future research.

Chapter 2

Gamma Deployment

In this chapter, we incrementally build a description of the study object of this dissertation. First, Section 2.1 presents the QoS metric named Gamma Deployment. Next, Section 2.2 applies this metric in the formal definition of Gamma Deployment Problem. Then, Section 2.3 explains a discretization approach to convert any instance of Gamma Deployment Problem into an instance of Gamma Deployment Problem in Grids.

2.1 Gamma Deployment Metric

Developed by Silva et al. [25], Gamma Deployment, or $\Gamma_D\binom{\tau}{\rho}$, is a QoS metric to establish project goals and evaluate performance of vehicular networks. According to Silva et al. [25], it can be applied to address problems such as the traffic flow monitoring and smoothness guarantee in Internet media consumption.

Gamma Deployment parameterizes a desired minimum share of vehicles within the road network that must be offered a minimum V2I inter-contact regularity. Definition 1 specifies Gamma Deployment in mathematical terms.

Definition 1 (Gamma Deployment Metric) *Assume a deployment R of RSUs in a road network M and a set T with all vehicles trips through M during a specific period of time. Let $T' \subseteq T$ be a set with all vehicle trips meeting at least one RSU during each τ-second time interval in their trips. Let ρ be the required percentage of covered vehicles. R is $\Gamma_D\binom{\tau}{\rho}$ if $\frac{|T'|}{|T|} \geq \rho$.*

A direct use of Definition 1 consists in setting up a pair (τ, ρ) to compose a $\Gamma_D\binom{\tau}{\rho}$ requirement for the project of a vehicular network. Based on this design constraint, any optimization method can minimize the number of RSUs deployed on a road network.

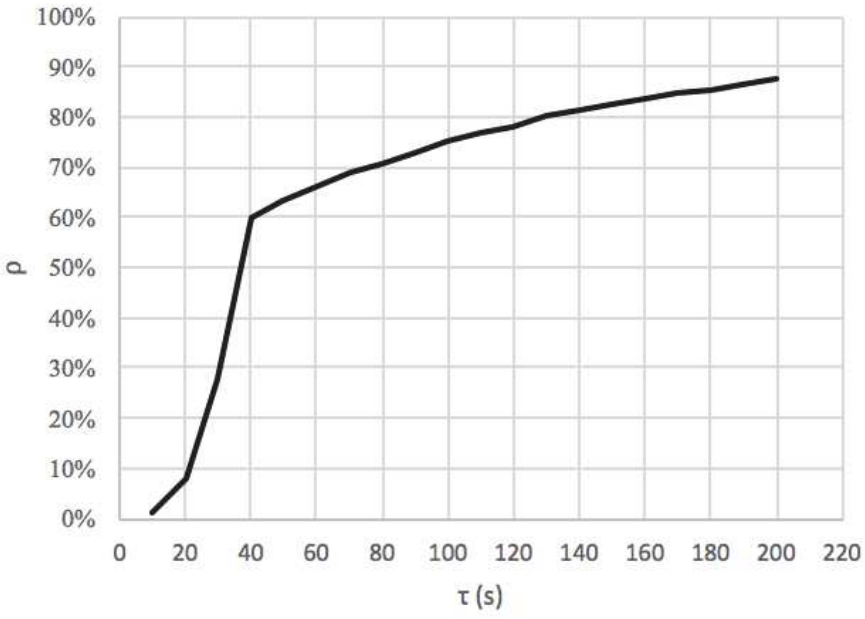


Figure 2.1: Characteristic Γ_D curve of a given deployment of RSUs. [8]

Notice that τ and ρ are independent parameters from this perspective, meaning that a designer of a vehicular network can choose an arbitrary combination of τ and ρ according to his or her design goals, such as $\Gamma_D\binom{40}{0.5}$, $\Gamma_D\binom{40}{1.0}$, $\Gamma_D\binom{10}{1.0}$, or any other.

Another application of Definition 1 consists in obtaining a Γ_D curve characterizing a specific deployment of RSUs on a road network. From this perspective, ρ is a dependent variable and τ keeps being an independent variable. Figure 2.1 exemplifies a Γ_D curve. The horizontal axis represents different inter-contact times τ. For a value of τ, the vertical axis ρ indicates the total percentage of vehicles meeting at least one RSU each τ-second time interval of their trips. To build this graph, consider a range $\{\tau|\tau = 20y, y \in \{1, 2, ..., \mu\}\}$ of values for τ. For each of them, calculate the percentage of covered vehicles, which is the corresponding value of ρ in the vertical axis. In Figure 2.1, the deployment is, at the same time, $\Gamma_D\binom{20}{0.1}$, $\Gamma_D\binom{40}{0.6}$, $\Gamma_D\binom{80}{0.7}$, and many other combinations.

2.2 Gamma Deployment Problem

Even though Gamma Deployment was originally proposed as a quality of service metric, its application to allocate RSUs in a vehicular network leads to the definition of combinatorial optimization problems. One of these combinatorial problems aims at

minimizing the number of deployed RSUs meeting the Gamma Deployment metric. We call this problem the Gamma Deployment Problem or GDP. In this section, we provide a formal definition for GDP using concepts of graph theory.

Let a walk W be a sequence of vertices of a graph such that there is an edge between each pair of consecutive vertices. Furthermore, let $L(W)$ be the number of vertices (distinct or not) in the walk W and let $W[j]$ be the vertex in the position j of W, with $j \in \{1, \ldots, L(W)\}$. For a walk W and a set of vertices R, let $W \setminus R$ be a collection containing all the maximal nonempty subwalks of W with no vertex in R. For example, $(1, 7, 2, 3, 1, 5, 4, 5, 6, 7) \setminus \{1, 5\} = \{(7, 2, 3), (4), (6, 7)\}$.

Definition 2 (Decision Version of GDP) *Let $G = (V, E)$ be a graph and let $T = \{T_1, \ldots, T_p\}$ be a collection of walks in G. Let $F : T \times \mathbb{Z}_+^* \to \mathbb{R}_+^*$ be a function associating a duration to each step j of each walk $T_i \in T$. Let $\tau \in \mathbb{R}_+^*$, $\rho \in [0, 1]$, and $k \in \mathbb{Z}_+^*$ respectively be the inter-contact time, the minimum required coverage, and the maximum allowed number of RSUs. Determine whether or not there exists a set $R \subseteq V$ such that $|R| \leq k$ and, for at least a fraction ρ of the walks $T_i \in T$, the following inequalities hold:*

$$\sum_{c=1}^{L(C)} F(C, c) < \tau \qquad \forall C \in T_i \setminus R \tag{2.1}$$

Figures 2.2a and 2.2b show an instance of GDP to exemplify Definition 2.

Definition 2 implies a way to check whether or not a subset R of V certifies that the given instance of the problem is a YES instance. This consists in verifying whether each $C \in T_i \setminus R$ satisfies $\sum_{c=1}^{L(C)} F(C, c) < \tau$. It can run in polynomial time since the number of elements in $T_i \setminus R$ is upper bounded by $L(T_i)$. Therefore, it is clear that the decision version of GDP belongs to NP.

2.3 Discretization of Road Networks

Although Definition 2 does not assume a specific topology for graph G, GDP has always been tackled for instances in which G is a grid [8, 9, 25–27]. For that reason, this work especially focuses on the Gamma Deployment Problem in Grids, or GDPG. In this section, we explain the discretization approach usually applied on road networks as a preprocessing step to overcome the complexity of deploying RSUs with continuous coordinates.

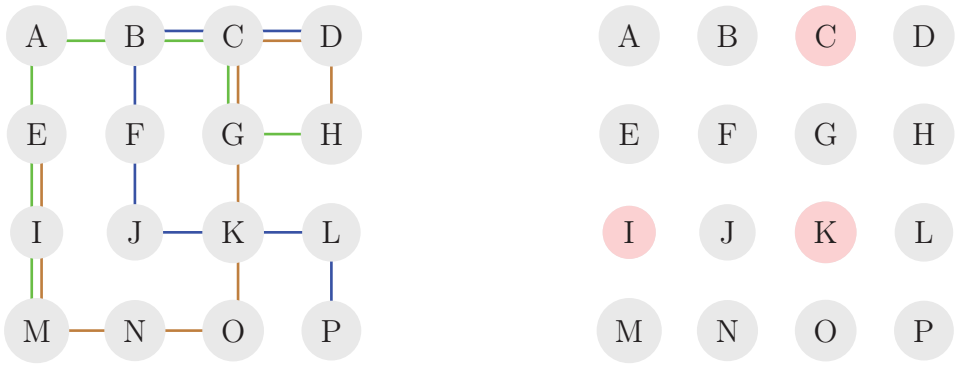
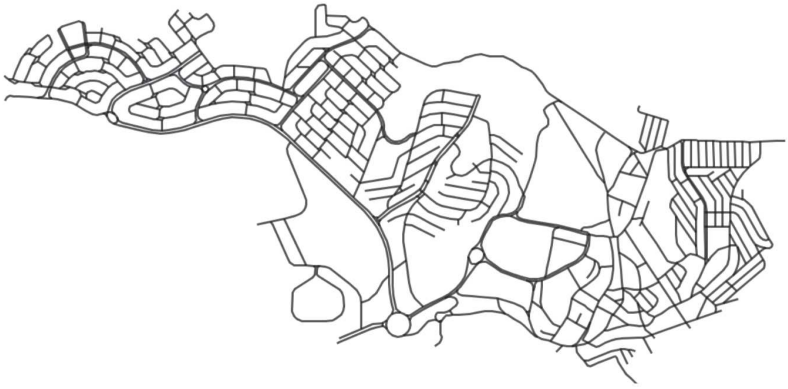


Figure 2.2: (a) An instance of GDP with three walks (represented in different colors), with $F(T_i, j) = 1$ for each step j of each walk T_i, and with the parameters $\tau = 4$ and $\rho = 100\%$. (b) An optimal solution for this instance: RSUs in the vertices C, I, and K.

First, the road network is partitioned into a matrix M of dimension $\psi \times \psi$. We say that each cell of M is an urban cell. The physical dimensions of the urban cells can be arbitrarily chosen based on the specific needs of the designer of the vehicular network. Usually, we choose the value of ψ so that each urban cell approximately represents the covered area in case an RSU is deployed on its center. Figure 2.3 illustrates this discretization approach.

It is important to mention that, even though we assume each urban cell is covered by deploying an RSU on it, an urban cell is not an exact reference for deployment position and covered area. There is a vast range of practical situations to address when determining the actual deployment coordinates, such as energy supply, topology, interference, and so forth. Moreover, the rectangular shape of an urban cell does not accurately represent the area covered by an RSU, which tends to be similar to a circle. Some works, more concerned about properly representing the coverage area of sensors and their communication patterns, modeled urban cells with hexagonal shape [1]. Nevertheless, previous works on GDPG considered a rectangular shape for the urban cells, which is also a good enough model for the purposes of our work.



(a)

0	38	0	0	274	17	0	209	348	0	0	0	0	0	0	0	0	0	0	0
0	180	595	124	1306	44	0	496	3107	276	0	0	0	0	0	0	0	0	0	0
420	1258	3018	2617	7351	0	130	5126	5765	8567	939	7901	3222	0	0	0	0	0	0	0
767	649	1573	4281	3867	9791	2551	6839	2040	651	6930	39	5669	402	0	0	0	0	0	0
936	2263	2947	2132	3708	5076	6961	5121	2965	879	277	307	0	4834	0	0	0	0	0	0
2	2312	5413	684	950	13539	4731	1318	1923	2130	416	1195	0	5413	1021	62	0	0	0	0
0	0	2772	8237	8902	482	14316	1943	771	4631	4981	3850	92	1728	3762	115	28	0	0	0
0	0	0	0	0	0	1529	9473	1901	5242	1289	1929	595	571	1490	5860	4767	1648	370	1
0	0	0	0	0	0	85	10099	1640	1241	1623	4305	3803	0	1457	975	2099	468	45	0
0	0	0	0	0	0	0	1692	10240	1672	1754	1698	4405	0	1817	550	1151	1008	0	0
0	0	0	0	0	0	0	194	10800	4795	3227	3154	4439	0	1560	1111	757	989	89	0
0	0	0	0	0	0	317	73	2562	7762	275	2309	4110	5252	2560	2562	546	782	90	0
0	0	0	0	0	0	129	84	0	7553	21	2528	0	35	5515	1610	545	795	408	0
0	0	0	0	0	0	78	1046	1	8745	0	6819	385	1711	1676	804	2070	944	364	0
0	0	0	0	0	0	0	380	1105	7214	323	4498	4829	3422	689	2283	933	859	30	0
0	0	0	0	0	0	7	41	911	6688	12326	5242	1752	115	2356	616	2505	644	0	0
0	0	0	0	0	0	7	30	1548	9755	3963	1149	3619	441	1051	3039	814	201	0	0
0	0	0	0	0	0	0	0	354	1194	388	0	1630	4338	3418	855	139	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	178	528	79	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0

(b)

Figure 2.3: (a) Layout of the road network of the city Ouro Branco, Brazil. (b) 20×20 discritized version of the road network of the city Ouro Branco, Brazil. [25]

Chapter 3

Proof of NP-Completeness

In this chapter, we demonstrate that the decision version of Gamma Deployment Problem in Grids is NP-complete. This proof implies that the decision version of Gamma Deployment Problem for general graphs is also NP-complete.

The remainder of this chapter is organized as follows: in Section 3.1, we briefly present the proving strategy used and provide precise definitions for the intermediary problems handled; in Section 3.2, we explicit the proofs.

3.1 Proving Strategy

Our strategy starts by proving an intermediary NP-completeness result, which the readers might find relevant on its own. That intermediary proof starts from the vertex cover problem in planar graphs with no degree higher than 3 (VCPPG3), which is NP-complete [10].

Definition 3 (Decision Version of VCPPG3) *Let $G = (V, E)$ be a planar graph in which there is no vertex with degree higher than 3 and let k be a positive integer number. Determine whether or not there exists a set $R \subseteq V$ such that all edges in E have at least one end-point in R and $|R| \leq k$.*

Starting from a general instance of VCPPG3, we perform a polynomial transformation to an instance of the cover of paths by vertices problem in grids (CPVPG). The CPVPG consists in a special case of the set covering problem in which the candidate elements are vertices of a grid and all the sets are simple paths in that grid. To the best of our knowledge, this problem has not yet been proposed in the literature.

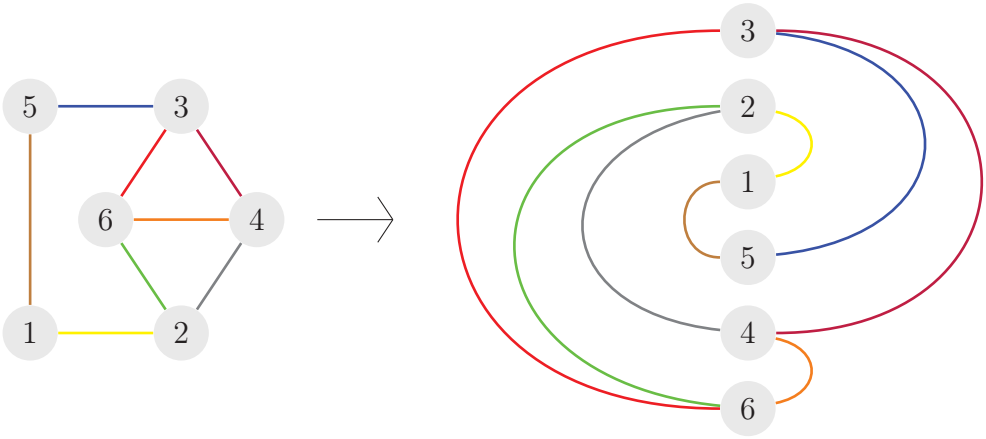


Figure 3.1: Embedding of a planar graph with no degree higher than 3 in a 2-page book.

Definition 4 (Decision Version of CPVPG) *Let $G = (V, E)$ be a grid, let S be a collection of simple paths in G, and let k be a positive integer number. Determine whether or not there exists a set $R \subseteq V$ with a non-empty intersection with each path in S, and such that $|R| \leq k$.*

In the final step of the proof, we perform a polynomial transformation from a generic instance of CPVPG to an instance of GDPG.

3.2 The Proofs

In this section, we prove that GDPG is NP-complete.

Lemma 1 *The decision version of CPVPG is NP-complete.*

Proof Given a planar graph $G = (V, E)$ with no degree higher than 3 and a positive integer number k, we build an instance of CPVPG composed of a grid G', a collection S of paths, and a positive integer number k'. The construction will ensure the existence of a vertex cover in G with cardinality not larger than k if, and only if, there exists a cover of the paths in S by vertices in the graph $G' = (V', E')$ with cardinality not larger than k'. Let $V = \{1, ..., n\}$ be the set of vertices of G and let $|E| = m$ be the number of edges in G.

In a book embedding of a graph $G = (V, E)$, we linearly dispose all its vertices in the spine of a book and sort them in a specific order, which is given by a function $F_V : V \to \mathbb{Z}_+^*$. Furthermore, we assign each edge of the graph to a specific page of the book according to a function $F_E : E \to \mathbb{Z}_+^*$. A page consists of a half-plane

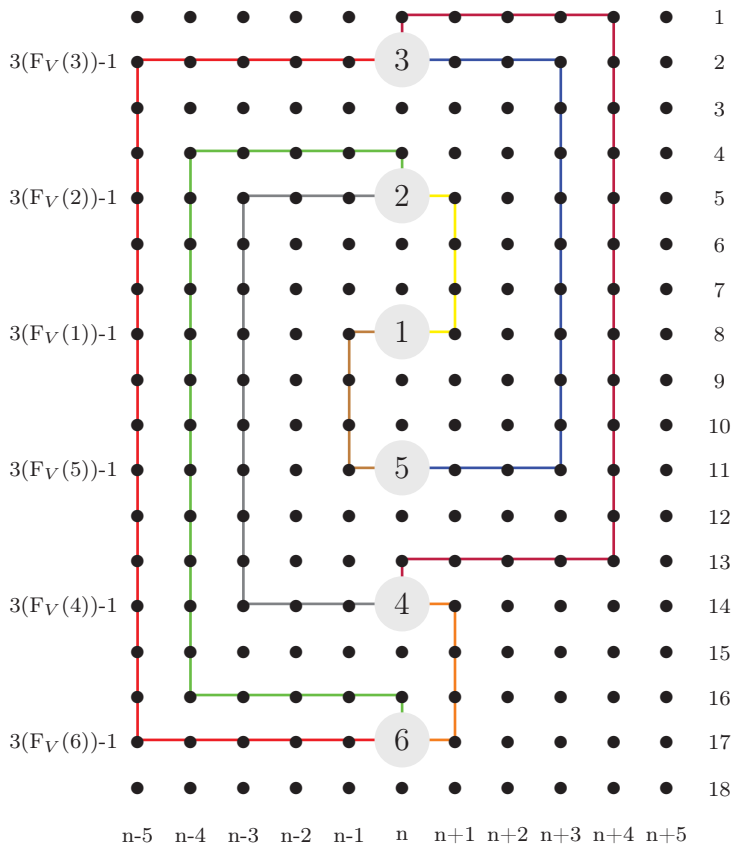


Figure 3.2: Vertices of G' and the collection of paths obtained from the two-page book embedding of G.

starting at the spine. All the edges assigned to a page are contained in its half-plane and do not intersect with each other. Heath [11] showed that all planar graphs with no degree higher than 3 can be embedded in a 2-page book. Then, the first step of our transformation consists in embedding G in a 2-page book, as Figure 3.1 exemplifies. This embedding can be done in polynomial time [4, 11].

Let $G' = (V', E')$ be a grid graph with $V' = \{1, ..., 3n\} \times \{1, ..., 2n - 1\}$, and $E' = \{((a, b), (c, d)), \forall (a, b), (c, d) \in V' : (|a - c| = 1 \wedge b = d) \vee (a = c \wedge |b - d| = 1)\}$. Let $H \subset V'$ be associated with the vertices of G following the order given by F_V, such that $H = \{(3(F_V(v)) - 1, n), \forall v \in V(G)\}$.

Then, we construct a collection $S = \{S_1, \ldots, S_m\}$ with m paths, each of them being a subgraph of G' and corresponding to one of the edges in E. Each path starts and ends in vertices belonging to the set H which represent the vertices of G joined by the edge corresponding to the path. We need, as we will later show, to assure that different paths do not intersect in vertices not in H. To ensure this, we construct each path from $(a, b) \in H$ to $(c, d) \in H$ according to the following rules:

1. Each path has a vertical component. If the corresponding edge $e \in E$ belongs to the left page, that component passes by vertices of the column $n - \frac{|b-d|}{3}$. Otherwise, by vertices of the column $n + \frac{|b-d|}{3}$. (For example, the red edge in Figure 3.1 belongs to the left page. The vertical component of the corresponding red path in Figure 3.2 passes by vertices of the column $n - \frac{|17-2|}{3} = n - 5$);

2. For each vertex $v \in V$, at least one of the pages in the book embedding contains either zero or one of its incident edges since the degree of v is at most 3. If there is an edge incident to v in that page, the corresponding path in S has a horizontal component in the row $3(f_V(v)) - 1$ from the column n to the column determined by rule 1. (For example, vertex 3 in Figure 3.1 has only one edge in its the left page: the red one. In Figure 3.2, the corresponding red path has a horizontal component in row $3(f_V(3)) - 1 = 2$ from the column n to the column $n - 5$);

3. For some vertices $v \in V$, there is a page in which v has either 2 or 3 incident edges. In this case, each of the corresponding paths must leave vertex $(3(f_V(v))-1, n)$ in a different direction (up, down, or horizontal). To decide which path should leave in which direction we take into account the order of the destination vertices in the spine of the book. This decision is taken in order to avoid crossings between the paths. (For example, vertex 6 in Figure 3.1 has 2 edges in its left page and both destination vertices are above it in the spine of the book. In Figure 3.2, the path to the farthest vertex (the red path) leaves vertex 6 in the horizontal direction and the path to the closest vertex (the green path) leaves v in the up direction. After one vertex in the up or horizontal direction, the path continues horizontally until it meets its vertical component built-in rule 1, i.e the red path goes until column $n - 5$ and green path goes until column $n - 4$).

Finally, we set $k' = k$.

Following rules 1, 2, and 3, we prove by contradiction that, for each pair of paths $S_1, S_2 \in S$, their intersection can only occur in the end vertices, which means $S_1 \cap S_2 \subseteq H$. Assume that there are two paths, $S_1 = (u_1, \ldots, u_2)$ and $S_2 = (w_1, \ldots, w_2)$, which intersect in a vertex not present in H. Note that $F_E((u_1, u_2)) = F_E((w_1, w_2))$. Assume, without loss of generality, that $F_V(u_1) < F_V(u_2)$, $F_V(w_1) < F_V(w_2)$, and $F_V(u_1) \leq F_V(w_1)$. We know that each end vertex of any path corresponds to a vertex in V. For simplification reasons, we will use the same symbols (u_1, u_2, w_1, and w_2) to refer to each respective vertex in V' and in V. Consider the case in which an end vertex of S_1 coincides with an end vertex of S_2. Rules 2 and 3 assure that the horizontal fragments of S_1 and S_2 always occur in different rows around the row containing its

corresponding end vertex, which is in H. Note that the path with a farthest destination among S_1 and S_2 has its vertical component on column farther from the n^{th} column. Thus, by rule 3, S_1 and S_2 cannot cross at any point not in H. Therefore, u_1, u_2, w_1, and w_2 must be distinct vertices. It is not possible that $F_V(u_2) < F_V(w_1)$ since the rows used by path S_1 would be strictly above those use by S_2. In case $F_V(u_1) < F_V(w_1) < F_V(w_2) < F_V(u_2)$, intersection cannot occur due to rule 1. Finally the case in which $F_V(u_1) < F_V(w_1) < F_V(u_2) < F_V(w_2)$ is not possible since the corresponding edges of E must cross each other and then they cannot be on the same page in the book embedding. Hence, we conclude that S_1 and S_2 cannot cross in vertices not in H.

Figure 3.2 illustrates V' and all the paths in S obtained by performing the rules 1, 2, and 3 on the graph obtained with the transformation shown in Figure 3.1.

Next, we prove that the answer for VCPPG3 on instance (G, k) is YES if, and only if, the answer for the CPVPG on instance (G', S, k') constructed by the above procedure is also YES.

Let R be a subset of V with $|R| \leq k$ which is a vertex cover for G. Take R' as the elements of H associated with the elements in R and note that since R is a cover of all edges in E, R' is a cover of all paths in S. Moreover, $|R'| = |R| \leq k = k'$. Hence, (G', S, k') is a YES instance.

Now, let $R' \subseteq V'$ with $|R'| \leq k'$ be a cover of the paths in S. With the construction of (G', S, k'), we made sure that all vertices in $V' \setminus H$ belong to at most one path in S. Thus, if there is a path $S_i \in S$ with $S_i \cap (R' \setminus H) \neq \emptyset$, we know that each element $u \in S_i \cap (R' \setminus H)$ covers no path other than S_i. Therefore, we can exchange them for a vertex in $H \cap S_i$ maintaining the cover. After repeating the above procedure exhaustively, we get a cover of paths by vertices $R'' \subseteq H$ with cardinality at most k'. Now, we can construct a vertex cover $R \subseteq V$ for G with $|R| \leq k$ by selecting each vertex in V associated with each element in R''. Then, (G, k) is a YES instance.

Finally, it is straightforward to prove that the CPVPG is in NP by inspection. □

We have just shown that CPVPG is NP-complete. This result is a premise in the proof of Theorem 1.

Theorem 1 *The decision version of GDPG is NP-complete.*

Proof Let $G = (V, E)$ be a grid with $V = \{v_{ab}, \forall (a, b) \in \{1, ..., m\} \times \{1, ..., n\}\}$ and $E = \{(v_{ab}, v_{cd}),\ \forall \{v_{ab}, v_{cd}\} \subseteq V : (|a - c| = 1 \wedge b = d) \vee (a = c \wedge |b - d| = 1)\}$. Let $S = \{S_1, ..., S_p\}$ be a collection of paths in G. Let $k \in \mathbb{Z}_+^*$.

Given the instance (G, S, k) of CPVPG, we obtain an instance $(G', T, F, \tau, \rho, k')$ of GDPG in the following way: $G' = G$, $T = S$, $k' = k$, $\tau = 1$, $\rho = 100\%$, and F such that $F(T_i, x) = \frac{1}{L(T_i)}, \forall x \in \{1, \ldots, L(T_i)\}$ and $\forall i \in \{1, \ldots, |T|)\}$.

Note that after this transformation, the following equality holds for each $T_i \in T$:

$$\sum_{c=1}^{L(T_i)} F(T_i, c) = 1 = \tau \tag{3.1}$$

Next, we prove that the answer for CPVPG on instance (G, S, k) is YES if, and only if, the answer for GDPG on instance $(G', T, F, \tau, \rho, k')$ constructed by the above procedure is also YES.

Let $R \subseteq V$ be a solution to the instance (G, S, k) of CPVPG. Note that R is a feasible deployment of RSUs complying with $\Gamma_D\binom{\tau}{\rho}$. Indeed, to satisfy inequalities (2.1) for each walk, at least one vertex on each walk should have an RSU and this holds since R covers all elements in $S = T$. A symmetric argument shows that if R satisfies $\Gamma_D\binom{\tau}{\rho}$ it is a cover of the paths in S.

Since GDPG belongs to NP, by the above argument it also belongs to NP-complete. □

As consequence of the proof that GDPG is NP-complete, we state Corollary 1.

Corollary 1 *The decision version of GDPG with $\tau = 1$ and $\rho = 100\%$ is NP-complete.*

This corollary is highly relevant since that special case ($\tau = 1$ and $\rho = 100\%$) was specifically tackled in [9, 25, 26] with heuristic and exponential exact approaches. We also deal with this special case of the problem later in this dissertation.

Chapter 4

Mixed Integer Linear Programming Formulations

In this chapter, we present and discuss integer linear programming formulations for Gamma Deployment Problem in Grids. In Section 4.1, we explain the multi-flow formulation proposed by [26]. In Section 4.2, we show a small issue in this formulation and provide a correction for it. In Section 4.3, we propose a set covering formulation and provide a proof that the dual bounds associated with its linear programming relaxation cannot be worse than the dual bounds associated with the linear programming relaxation of the multi-flow formulation.

4.1 Multi-Flow Formulation

The multi-flow ILP formulation proposed by Silva et al. [26] starts with a preprocessing procedure to build a directed graph $H_i = (V_i, A_i)$ associated to each walk $T_i \in T$. The vertices in V_i are those vertices of V visited by T_i plus two artificial vertices: a source s and a sink t. Vertex s is a predecessor of all vertices $T_i[j]$ such that $\sum_{r=1}^{j-1} F(T_i, r) < \tau$. Vertex t is a successor of all vertices $T_i[j]$ such that $\sum_{r=j+1}^{L(T_i)} F(T_i, r) < \tau$. Finally, there is an arc in A_i from vertex $T_i[j]$ to vertex $T_i[k]$ if, and only if, $j < k$ and $\sum_{r=j+1}^{k-1} F(T_i, r) < \tau$. Figure 4.1 illustrates this preprocessing method.

Let $\delta_i^+(a)$ be the set with all the indexes w such that $(a, T_i[w])$ is an arc of the graph H_i. Let $\delta_i^-(b)$ be the set with all the indexes w such that $(T_i[w], b)$ is an arc of the graph H_i. Furthermore, consider the following decision variables:

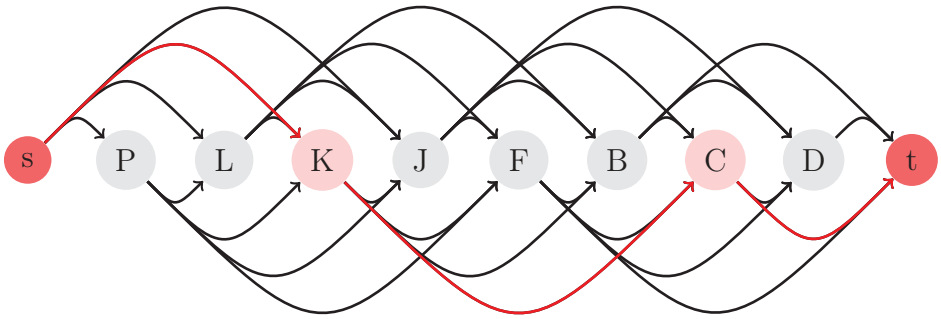


Figure 4.1: Directed graph H_i obtained by applying the preprocessing procedure of the multi-flow ILP formulation on the blue walk in Figure 2.2a. For the chosen solution, the flow goes from vertex s to vertex K, from vertex K to vertex C, and from vertex C to vertex t.

$$c_a = \begin{cases} 1 & \text{if an RSU is deployed in } a \in V; \\ 0 & \text{otherwise.} \end{cases}$$

$$z_i = \begin{cases} 1 & \text{if the walk } T_i \text{ is covered;} \\ 0 & \text{otherwise.} \end{cases}$$

$$f_{ab}^i = \begin{cases} 1 & \text{if the arc } (a,b) \text{ from } A_i \text{ is used in the path from } s \text{ to } t; \\ 0 & \text{otherwise.} \end{cases}$$

The path induced by variables f_{ab}^i implies how the walk T_i is covered: the arcs go from s to t visiting only intermediary vertices that contain an RSU. The multi-flow ILP formulation follows:

$$\min \sum_{a \in V} c_a \tag{4.1}$$

Subject to

$$\sum_{k \in \delta_i^+(s)} f^i_{s,T_i[k]} = z_i \quad \forall i | T_i \in T \tag{4.2}$$

$$\sum_{k \in \delta_i^-(t)} f^i_{T_i[k],t} = z_i \quad \forall i | T_i \in T \tag{4.3}$$

$$\sum_{k \in \delta_i^-(l)} f^i_{T_i[k],l} - \sum_{k \in \delta_i^+(l)} f^i_{l,T_i[k]} = 0 \quad \forall i | T_i \in T, \forall l \in V_i \setminus \{s,t\} \tag{4.4}$$

$$f^i_{al} \leq c_a \quad \forall i | T_i \in T, \forall (a,l) \in A_i | a \notin \{s,t\} \tag{4.5}$$

$$f^i_{la} \leq c_a \quad \forall i | T_i \in T, \forall (l,a) \in A_i | a \notin \{s,t\} \tag{4.6}$$

$$\sum_{i | T_i \in T} z_i \geq \rho |T| \tag{4.7}$$

$$c_a \in \{0,1\} \quad \forall a \in V \tag{4.8}$$

$$z_i \in \{0,1\} \quad \forall i | T_i \in T \tag{4.9}$$

$$f^i_{ab} \in \{0,1\} \quad \forall (a,b) \in A_i, \forall i | T_i \in T \tag{4.10}$$

The objective function (4.1) minimizes the number of vertices chosen from V to deploy RSUs. Constraints (4.2) state that a path starts in vertex $s \in V_i$ for each covered walk i. Equivalently, constrains (4.3) state that a path ends in the vertex $t \in V_i$ for each covered walk i. Constraints (4.4) guarantee flow conservation for all vertices of H_i except s and t. Constraints (4.5) and (4.6) ensure that flow is only possible to exist in an arc if both its ends are associated to vertices chosen to have an RSU. Finally, Constraint (4.7) guarantees a minimum percentage coverage ρ of walks.

Observe that only the decision variables c_a and z_i have meaning regarding GDPG, while variables f^i_{ab} do not. Moreover, the flow from s to t in a graph H_i does not necessarily reach all vertices whose corresponding variable c_a equals 1. In other words, for a given feasible solution for GDPG, there can be a huge number of different paths from s to t in each graph H_i. It implies an excessively large number of feasible integer solutions, which leads to poor performance in commercial optimizers. Another perspective leading to the same conclusion consists in noticing the Objective Function (4.1): flow and multi-flow formulations tend to be weak and slow when the objective function only depends on the chosen vertices.

4.2 Issue with the Multi-Flow Formulation

In this section, we show an issue and suggest a small correction for the ILP formulation of Gamma Deployment Problem in Grids (GDPG) proposed by Silva et al. [26], which is based on multi-flow. The vertex definition for H_i works well when all the walks in T are paths, but it may introduce errors otherwise.

An objection could be made to the word "correction". Apparently, the term "extension" could be more appropriate since Silva et al. [26] never explicitly mentioned the possibility of cycles in some of the vehicle trips. Nevertheless, they used the discretization approach explained in Section 2.3, which implies that even paths in the original road network can become cyclic walks after the discretization.

If a vertex $w \in V_i$ is visited at least twice in a walk T_i, an arc $(v, w) \in A_i$ may represent that the inter-contact time is respected from the first visit of vertex v to the first visit of vertex w. But, wrongly, it can also represent that the inter-contact time is respected from the first visit of vertex v to the last visit of vertex w.

Figure 4.2 exemplifies why the multi-flow formulation does not behave correctly with cyclic walks. In this example, the graph H_i generated by the preprocessing step is not an acyclic direct graph as stated in [26]. A feasible path from s to t in H_i is (s, B, t). This path implies that the corresponding walk T_i would be covered if only the urban cell B had an RSU, which is clearly false.

We propose a slight modification to fix this issue. It simply consists of defining each vertex of H_i as an ordered pair containing the identifier of the corresponding vertex in G and the order in which it appears in the walk T_i, that is, $V_i = \{(T_i[j], j), j \in \{1, \ldots, L(T_i)\}\} \cup \{s, t\}$. Now, arcs can be defined accordingly and all graphs H_i will be directed acyclic graphs. In the formulation, variables c_a and z_i do not change and variables f^i_{ab} become:

$$f^i_{(T_i[j],j),(T_i[k],k)} = \begin{cases} 1 & \text{if arc } ((T_i[j], j), (T_i[k], k)) \in A_i \text{ is used in the path from } s \text{ to } t; \\ 0 & \text{otherwise.} \end{cases}$$

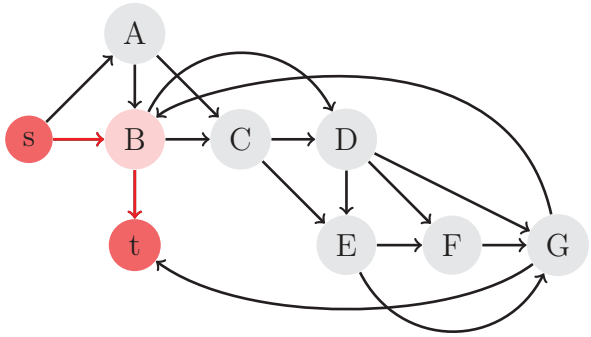
The Objective Function (4.1) is kept the same. Constraints (4.2), (4.3), and (4.4) are just adapted to reflect flow conservation in the new graphs H_i. Constraint (4.7) does not change. Constraints (4.5), and (4.6) are respectively replaced by:

$$f^i_{(T_i[j],j),(T_i[k],k)} \leq c_{T_i[j]} \quad \forall i | T_i \in T, \forall ((T_i[j], j), (T_i[k], k)) \in A_i | a \notin \{s, t\} \quad (4.11)$$

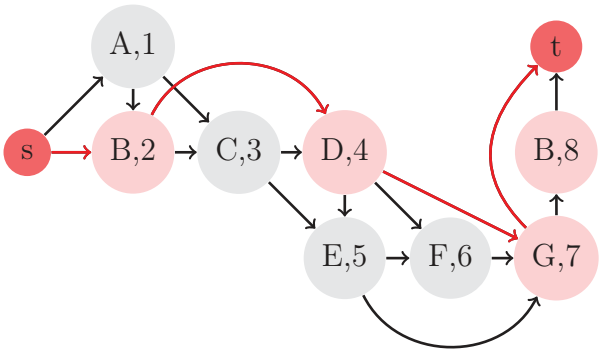
$$f^j_{(T_i[k],k),(T_i[j],j)} \leq c_{T_i[j]} \quad \forall i | T_i \in T, \forall ((T_i[k], k), (T_i[j], j)) \in A_i | a \notin \{s, t\} \quad (4.12)$$

k	1	2	3	4	5	6	7	8
$F(T_i, k)$	5	13	12	10	5	7	16	6
$T_i[k]$	A	B	C	D	E	F	G	B

(a) Walk



(b) Original Formulation



(c) Corrected Formulation

Figure 4.2: A cyclic walk T_i and the corresponding graphs H_i in the original multi-flow formulation and in the corrected multi-flow formulation for $\tau = 15$. In the original formulation, a path from s to t in H_i is (s, B, t), which implies a wrong result. In the corrected formulation, (s, B, t) is not a feasible path from s to t and all the actually feasible paths from s to t imply correct coverages of T_i.

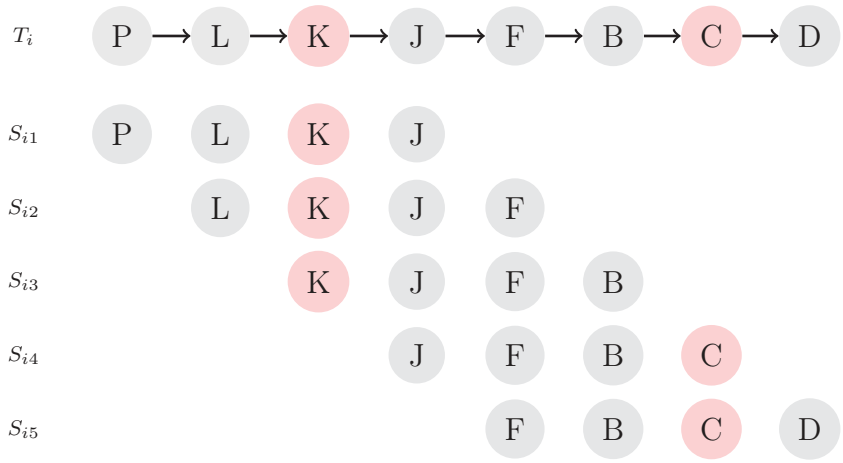


Figure 4.3: Sets S_{ij} obtained by applying the preprocessing procedure of the set covering ILP formulation on the blue walk in Figure 2.2a. For the chosen solution, all these sets are covered.

4.3 Set Covering Formulation

In this section, we propose a new ILP formulation for GDPG based on set covering.

A preprocessing procedure is used to build a collection S whose elements are sets $S_{ij} \subseteq V$. For a walk $T_i \in T$ in which $\sum_{k=1}^{L(T_i)} F(T_i, k) \geq \tau$, build all minimal sets $S_{ij} = \{T_i[k] | k \in \{j, \ldots, y\}, y \leq L(T_i)\}$ satisfying the inequalities $\sum_{k=j}^{y} F(T_i, k) \geq \tau$. Figure 4.3 illustrates this preprocessing procedure.

In case walk T_i is covered, each set S_{ij} must contain at least one vertex in which an RSU is deployed by definition of GDPG.

Consider the following decision variables:

$$c_a = \begin{cases} 1 & \text{if an RSU is deployed in } a \in V; \\ 0 & \text{otherwise.} \end{cases}$$

$$z_i = \begin{cases} 1 & \text{if the walk } T_i \text{ is covered;} \\ 0 & \text{otherwise.} \end{cases}$$

Then, a new ILP formulation for GDPG follows:

$$\min \sum_{a \in V} c_a \tag{4.13}$$

Subject to

$$\sum_{a \in S_{ij}} c_a \geq \quad z_i \quad \forall S_{ij} \in S \tag{4.14}$$

$$\sum_{i|T_i \in T} z_i \geq \quad \rho|T| \tag{4.15}$$

$$c_a \in \quad \{0,1\} \quad \forall a \in V \tag{4.16}$$

$$z_i \in \quad \{0,1\} \quad \forall i|T_i \in T \tag{4.17}$$

Objective function (4.13) minimizes the number of vertices chosen from V to deploy RSUs. Constraints (4.14) ensure that each set S_{ij} associated with a covered walk T_i is covered by at least one of the selected vertices. Constraint (4.15) ensures that at least a percentage ρ of the walks is covered.

Notice that the set covering formulation only uses a specific subset of variables from the multi-flow formulation: those with meaning for GDPG. Furthermore, it replaces constraints (4.2), (4.3), (4.4), (4.5), and (4.6) by constraints (4.14). From these observations, the set covering formulation is highly expected to be a more efficient and stronger formulation. With Theorem 2, we provide a formal proof that the dual bound associated with the linear programming relaxation of the set covering formulation is not weaker than the dual bound associated with the linear programming relaxation of the corrected multi-flow formulation.

Theorem 2 *The polytope associated with the linear programming relaxation of the set covering formulation is contained in the polytope associated with the linear programming relaxation of the corrected multi-flow formulation.*

Proof We prove that, given a solution in the polytope associated with the linear programming relaxation of the set covering formulation, there exists a corresponding solution in the polytope associated with the linear programming relaxation of the corrected multi-flow formulation with the same objective function value. Note that the variables c_a and z_i exist in both formulations with the same meaning, by which we can keep their values in both polytopes. It satisfies Constraint (4.7) (which is equal to Constraint (4.15)) and guarantees the same objective function value in both formulations. Assuming that constraints (4.14) and (4.15) hold, it suffices to show how to assign value to the variables f_{ab}^i in such way that the constraints (4.2), (4.3), (4.4), (4.11), and (4.12) hold.

Algorithm 1 presents a procedure to compute all the flows f_{ab}^i in a graph H_i given values of c_a and z_i that meet constraints (4.14) and (4.15). Its basic idea consists in

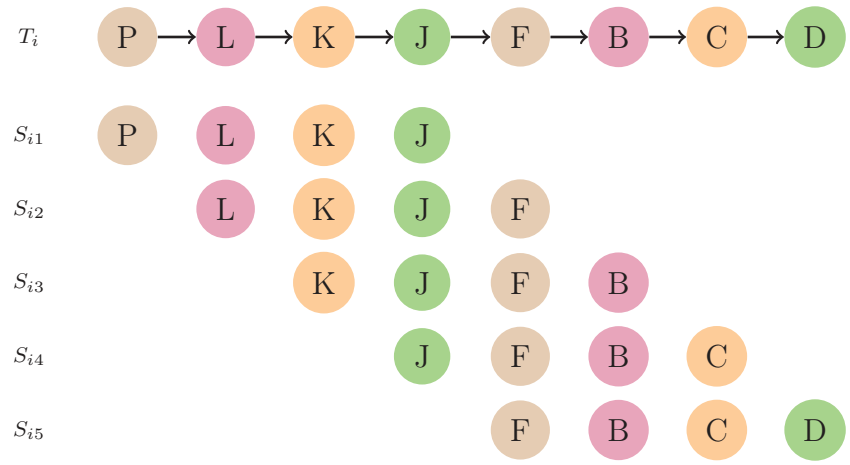
moving the flow z_i from s to t orderly passing between the vertices of H_i respectively associated with the sets S_{i1}, S_{i2}, ..., $S_{i\sigma}$ (in which $S_{i\sigma} \in S$ is the set S_{ij} with the greatest value of j). Figure 4.4 shows an input for Algorithm 1 and its corresponding output.

Input: i, z_i, c_a $\quad \forall a \in V_i$
Output: f^i_{ab} $\quad \forall (a,b) \in A_i$

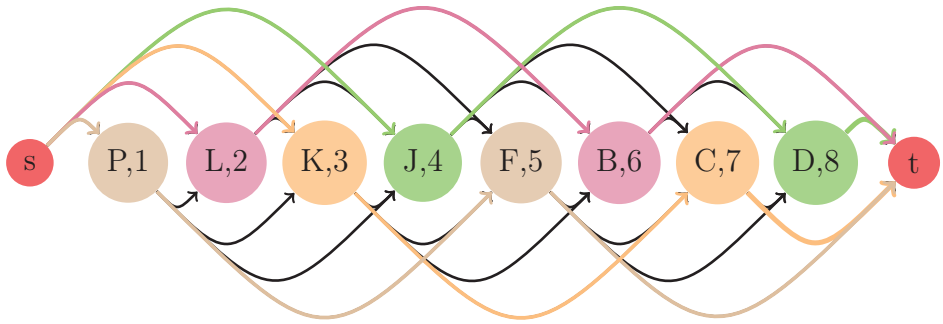
1 $tmp_j \leftarrow 0 \quad \forall j \in V_i$
2 $f^i_{ab} \leftarrow 0 \quad \forall (a,b) \in A_i$
3 $ideal_{jk} \leftarrow \frac{c_{T_i[k]}}{\sum_{r \in \delta^+_i((T_i[j],j))} c_{T_i[r]}} z_i \quad \forall j | \exists S_{i(j+1)} \in S, \forall k \in \delta^+_i((T_i[j],j))$
4 $f^i_{s,(T_i[j],j)} \leftarrow ideal_{0j} \quad \forall j \in \delta^+_i(s)$
5 $tmp_j \leftarrow ideal_{0j} \quad \forall j \in \delta^+_i(s)$
6 **foreach** $j | \exists S_{i(j+1)} \in S$ **do**
7 $D \leftarrow \delta^+_i((T_i[j],j)) \setminus \delta^+_i((T_i[j-1],j-1))$
8 $idealD_j \leftarrow \sum_{k \in D} ideal_{jk}$
9 $tmpD \leftarrow min(idealD_j, tmp_j)$
10 $tmp_j \leftarrow max(tmp_j - idealD_j, 0)$
11 $f^i_{(T_i[j],j),(T_i[k],k)} \leftarrow \frac{ideal_{jk}}{idealD_j} tmpD \quad \forall k \in D$
12 **foreach** $r \in \delta^+_i((T_i[j],j)) \setminus D$ **do**
13 $\Delta \leftarrow ideal_{jr} - tmp_r$
14 $tmp_r \leftarrow tmp_r + \Delta$
15 $f^i_{(T_i[j],j),(T_i[r],r)} \leftarrow f^i_{(T_i[j],j),(T_i[r],r)} + max(\Delta, 0)$
16 $tmp_j \leftarrow tmp_j - max(\Delta, 0)$
17 $f^i_{(T_i[r],r),(T_i[k],k)} \leftarrow \frac{ideal_{jk}}{idealD_j} max(-\Delta, 0) \quad \forall k \in D$
18 $tmpD \leftarrow tmpD + max(-\Delta, 0)$
19 $tmp_k \leftarrow \frac{ideal_{jk}}{idealD_j} tmpD \quad \forall k \in D$
20 $f^i_{(T_i[j],j),t} \leftarrow tmp_j \quad \forall j \in \delta^-_i(t)$

Algorithm 1: Assignment of feasible values for the variables f^i_{ab} for a walk T_i given a solution in the polytope of the set covering ILP formulation.

In row 3 of Algorithm 1, we define variables $ideal_{jk}$, whose values for a given j form a weighted distribution of the total network flow z_i exclusively among the successors $(T_i[k], k)$ of vertex $(T_i[j], j)$. In rows 4 and 5, we start by moving the flow z_i from s into all its successors according to variables $ideal_{0k}$, certifying Constraint (4.2) for the given i. To guarantee flow conservation, variables tmp_j represent the amount of flow present in vertex $(T_i[j], j)$ at each moment of the algorithm. (In Figure 4.4, see the flow through each arc with origin in vertex s. After the execution of row 4, $tmp_1 = 0.1$, $tmp_2 = 0.2$, $tmp_3 = 0.3$, and $tmp_4 = 0.4$)



(a) Noninteger Solution for Set Covering Formulation



(b) Equivalent Solution for Corrected Multi-flow Formulation

Figure 4.4: Example based on the blue walk in Figure 2.2a: noninteger solution in the polytope of the set covering formulation (input for Algorithm 1) and corresponding solution in the polytope of the multi-flow formulation (output from Algorithm 1). We use colors to represent the fractional values assigned to variables: (i) black represents 0; (ii) brown represents 0.1; (iii) purple represents 0.2; (iv) orange represents 0.3; and (v) green represents 0.4. The color of a vertex represents the value of the variable c_a associated to it. The color of an arc represents the value of the variable $f^i_{T_i[j],j,T_i[k],k}$ associated to it.

From row 6 to row 19, we establish a loop whose first iteration has $j = 1$ and last iteration has $j = \sigma - 1$. Variable D is a collection containing the indexes k of the vertices that are successors of $(T_i[j], j)$ but are not successors of $(T_i[j-1], j-1)$. We simulate an artificial vertex substituting all vertices contained in D. We define variables $idealD_j$ and $tmpD$ with analogous meaning with variables $ideal_{jk}$ and tmp_k respectively. The ideal flow of this artificial vertex among the other successors of $(T_i[j], j)$ is stored in $idealD_j$ and consists of the sum of the ideal flow of the real vertices it comprises. The temporary flow contained in this artificial vertex is stored in $tmpD$ and, in row 11, it is assigned to the corresponding actual flows. Variable Δ stores the difference between the ideal flow and the current flow in each successor $(T_i[r], r)$, $r \notin D$, of $(T_i[j], j)$. If $\Delta = 0$, the corresponding iteration of the loop in rows 12-18 changes nothing. If $\Delta > 0$, rows 15 and 16 attribute to it the missing flow coming from $(T_i[j], j)$. If $\Delta < 0$, rows 17 and 18 eliminate from it the exceeding flow, which goes to the vertices in D in a weighted fashion. In row 19, the total flow in the artificial vertex representing D also goes to the vertices in D in a weighted fashion. By the end of an iteration j of the outer loop, we ensure that the whole flow z_i of the network will be exclusively split between the successors $(T_i[k], k)$ of $(T_i[j], j)$ reproducing the respective values of $ideal_{jk}$. This loop ensures flow conservation as it is defined in constraints (4.4). (In Figure 4.4, $|D| = 1$ and $\Delta = 0$ for each iteration of the loop, resulting in a flow transferring the current tmp_j from $(T_i[j], j)$ to the vertex contained in D)

Finally, in row 20, we move the flow z_i from the vertices in $\delta_i^+((T_i[\sigma-1], \sigma-1)) = \delta_i^-(t)$ into t, satisfying Constraint (4.3) for i. (In Figure 4.4, each arc with destination in t fully transfers the flow the corresponding t predecessor to t)

For an acyclic walk T_i, note that (i) there is a one-to-one correspondence between S_{i1} and $\delta_i^+(s)$, (ii) there is a one-to-one correspondence between $S_{i(j+i)}$ and $\delta_i^+((T_i[j], j))$ for $1 \leq j \leq \sigma - 1$, and (iii) there is a one-to-one correspondence between the $S_{i\sigma}$ and $\delta_i^-(t)$. For cyclic walks, the above observations only differ in the fact that a single element in S_{ij} can be associated with multiple successors or predecessors of a single vertex in H_i. In either case, Constraint (4.14) for a given $S_{i(j+1)} \in S$ ensures that $ideal_{jk} \leq c_{T_i[k]}$, which ensures that constraints (4.11) and (4.12) are satisfied during the whole algorithm. □

Chapter 5

Computational Experiments

In this chapter, we show and analyze some experiments with the corrected multi-flow formulation and the set covering formulation.

The scope of this chapter is not to provide a complete comparison of both formulations from a practical perspective. This would require exhaustive experimentation with different datasets and tests with a variety of configurations in the optimizer. Instead, our computational experiments are aimed at illustrating the theoretical remarks expressed in Section 4.3 and showing some practical effects of running, within a limited time, the set covering formulation instead of the multi-flow formulation.

The rest of the chapter is organized as follows: in Section 5.1, we give some general characteristics about the dataset used and provide the parameters adopted to create our instances; in Section 5.2, we explicit the computational results and analyze them.

5.1 Dataset and Parameters

In our experiments, we used the pruned version of an urban mobility trace dataset based on the city of Cologne, Germany [32][1]. It consists of a realistic simulation of car traffic in a 400-km^2 area, comprising 75,515 individual vehicle trips over a period of 2 hours. We prepared the data with the preprocessing approach explained in Section 2.3, in which the road network was discretized into a 100×100 grid of urban cells and each vehicle trip was converted into a walk in this grid.

From the total of walks, we selected subsets with the first 100, 500, 1000, and 1500 walks to compose our instances. For each subset of walks, we generated a total of

[1]Mobility trace available at http://kolntrace.project.citi-lab.fr/

36 explicit instances for each formulation based on the criteria $\Gamma_D\binom{\tau}{\rho}$ with all possible value selections of the parameters $\tau \in \{40s, 80s, 120s\}$ and $\rho \in \{0.6; 0.8; 1.0\}$. The preprocessing time necessary to explicit both formulations was negligible for all the instances.

5.2 Results and Discussion

The experiments ran in a machine powered by an Intel Xeon E5405 2.00GHz, 15Gb of RAM, and Ubuntu operating system. We used the commercial optimizer CPLEX version 12.6 with standard configurations and a time limit of 1 hour for each execution. In total, there were 144 executions: for each of 36 different combinations of parameters, we ran an integer linear program and its corresponding linear programming relaxation for each formulation.

Table 5.1 shows the obtained results. The first three columns show the number of walks and the parameters τ and ρ. The next four columns show, respectively, the following pieces of information concerning the corrected multi-flow formulation: (i) the percentual linear programming relaxation gap; (ii) the objective function value of the best integer solution; (iii) the percentual final gap; and (iv) the running time, in seconds. The last four columns show the same information concerning the set covering formulation. We put a dash ('-') in some of the table cells to represent two situations: (i) there was no linear programming relaxation gap, which means the linear programming relaxation was not completely solved within the available time; or (ii) there was no final gap, which means an optimum solution was found and proved during the available time.

For instances with the same number of walks, the problem seems to become harder when τ increases or ρ decreases. In both cases, the problem becomes less constrained, admitting more feasible solutions of good quality. Hence, the optimal solutions for these instances tends to be harder to find.

The results show that the set covering formulation overwhelmingly outperformed the multi-flow formulation: (i) 3 instances were fully solved by both formulations; (ii) 28 instances were fully solved only by the set covering formulation; and (iii) 5 instances were not fully solved by any formulation. In the 3 instances solved by both formulations, the running time for the set covering one was at least 400 times shorter than for the multi-flow formulation. In most of the 28 instances solved only by the set covering formulation and in all the 5 instances not solved by any formulation, the best primal bound obtained with the multi-flow formulation is considerably higher than the best primal bound found with the set covering formulation. Furthermore, the final gaps

Table 5.1: Results for the multi-flow and the set covering ILP formulations.

Parameters			**Multi-flow formulation**				**Set covering formulation**			
$\|T\|$	τ	ρ	$LG(\%)$	**Z***	$FG(\%)$	t(s)	$LG(\%)$	**Z***	$FG(\%)$	t(s)
100	**40**	**0.6**	19.35	217	-	1136	1.84	217	-	<1
		0.8	17.68	311	-	414	0.96	311	-	<1
		1	18.51	470	-	633	0.64	470	-	<1
	80	**0.6**	30.23	134	8.71	3600	3.88	129	-	<1
		0.8	29.03	196	7.54	3600	2.69	186	-	<1
		1	29.89	285	4.20	3600	2.49	281	-	<1
	120	**0.6**	27.71	88	13.07	3600	3.61	83	-	<1
		0.8	27.87	126	8.25	3600	2.46	122	-	<1
		1	30.69	200	11.87	3600	2.12	189	-	<1
500	**40**	**0.6**	19.50	987	57.10	3600	2.04	441	-	5
		0.8	18.04	678	8.37	3600	1.71	643	-	3
		1	16.21	944	0.71	3600	1.27	944	-	<1
	80	**0.6**	29.15	693	69.31	3600	6.07	247	-	63
		0.8	26.18	807	60.95	3600	3.90	359	-	20
		1	24.44	2354	78.22	3600	2.96	540	-	<1
	120	**0.6**	28.10	532	76.96	3600	8.50	153	-	647
		0.8	28.76	636	70.61	3600	6.44	233	-	61
		1	25.83	2356	86.32	3600	3.33	360	-	<1
1000	**40**	**0.6**	18.75	1264	61.43	3600	2.21	544	-	32
		0.8	16.98	1544	51.98	3600	1.90	789	-	23
		1	14.27	1178	1.54	3600	1.27	1177	-	<1
	80	**0.6**	-	871	72.29	3600	-	294	0.67	3600
		0.8	26.38	1033	64.70	3600	4.59	436	-	136
		1	22.58	2708	76.27	3600	3.08	682	-	<1
	120	**0.6**	-	697	100	3600	-	180	2.04	3600
		0.8	25.93	809	100.00	3600	5.19	270	-	470
		1	24.05	2710	85.37	3600	4.01	449	-	<1
1500	**40**	**0.6**	19.29	2907	81.93	3600	3.38	591	-	3038
		0.8	16.78	2909	73.02	3600	1.95	870	-	49
		1	12.91	1324	2.23	3600	1.15	1309	-	<1
	80	**0.6**	-	2910	100.00	3600	-	322	2.28	3600
		0.8	25.84	2913	100.00	3600	4.41	476	-	669
		1	21.75	2929	75.33	3600	3.86	777	-	<1
	120	**0.6**	-	806	100.00	3600	-	195	3.63	3600
		0.8	-	930	100.00	3600	-	295	1.13	3600
		1	23.47	2930	97.34	3600	4.73	507	-	<1

for these instances at the time limit are much smaller for the set covering formulation than for the multi-flow formulation.

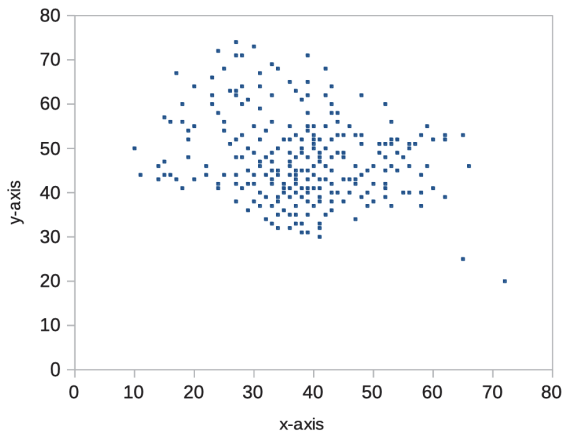
The theoretical result expressed by Theorem 2 opens two mutually exclusive possibilities: (i) both formulations share the same linear programming relaxation polytope; or (ii) the polytope associated with the set covering formulation is strictly contained in the polytope associated with the multi-flow formulation. The first hypothesis could only be proved theoretically or exhaustively by showing that the multi-flow formulation is contained in the set covering formulation. On the other hand, the second hypothesis is trivially true for instances whose linear programming relaxation bound differs from formulation to formulation. As the results show that the linear programming relaxation gap is considerably lower for the set covering formulation in all instances, therefore their polytopes are strictly contained in the corresponding polytopes of the multi-flow formulation.

The results concerning performance of the formulations and linear programming relaxation gaps were theoretically expected, as we saw in Chapter 4. Nevertheless, the multi-flow formulation was not designed with the objective of achieving a high performance. Instead, it was built to evaluate the deployments obtained with the Gamma-g heuristic for small instances, which it properly does. On the other hand, we designed the set covering formulation with the declared aim of improving performance as much as possible, and the results suggest successful completion of our purpose.

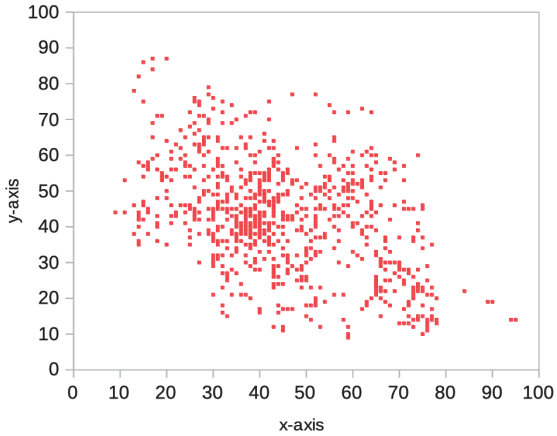
Concerning the deployment of RSUs obtained within one hour of running time, we analyze some instances with $|T| = 500$ and $\tau = 80s$. Figure 5.1 displays the physical distribution of RSUs in the deployments from both formulations for instances with $\rho = 0.6$ and $\rho = 0.8$. It compares deployments obtained with the multi-flow formulation and deployments obtained with the set covering formulation. The deployments from the set covering formulation are optimal and require fewer than half the number of RSUs. These reduced numbers of RSUs represent considerable financial savings for the designer of the vehicular network, since each RSU has installation and maintenance costs.

Figure 5.2 shows the characteristic Γ_D curves of the deployments represented in Figure 5.1. The required QoS metrics associated with Figure 5.2a and Figure 5.2b are, respectively, $\Gamma_D\binom{80}{0.6}$ and $\Gamma_D\binom{80}{0.8}$. In each of these figures, the blue curve (related to the set covering formulation) meets its respective QoS constraint in a tight way. On the other hand, the red curves (related to the multi-flow formulation) reach QoS marks beyond the requirements. In Figure 5.2a, besides being the $\Gamma_D\binom{80}{0.6}$, the red curve can be $\Gamma_D\binom{80}{0.77}$ and $\Gamma_D\binom{70}{0.6}$. It means that, keeping $\tau = 80s$, the corresponding deployment covers 77% of vehicles instead of 60% and, keeping $\rho = 0.6$, the corresponding

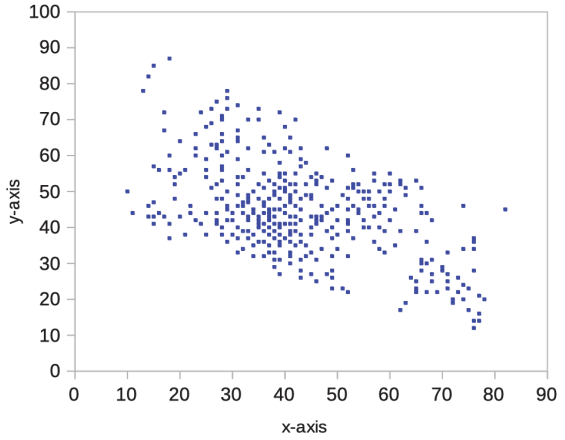
deployment allows a 10s-lower inter-contact time. In Figure 5.2b, besides meeting the $\Gamma_D\binom{80}{0.8}$ requirement, the red curve can also be $\Gamma_D\binom{80}{0.9}$ and $\Gamma_D\binom{75}{0.8}$. It means that, keeping $\tau = 80s$, the corresponding deployment covers 90% of vehicles instead of 80% and, keeping $\rho = 0.8$, the corresponding deployment allows a $5s$-lower inter-contact time. Nevertheless, it does not mean that the deployments from the multi-flow formulation were better. With the same number of RSUs used in the solutions from the multi-flow formulation, the designer of the vehicular network could significantly increase ρ keeping $\tau = 80s$ or decrease τ keeping $\rho \in \{0.6, 0.8\}$ if he or she used the set covering formulation with other parameters, as shown in Table 5.1. In this table, the solutions from the set covering formulation with QoS requirements $\Gamma_D\binom{80}{1}$ and $\Gamma_D\binom{40}{0.8}$ use much fewer RSUs than the multi-flow ILP solution with QoS requirement $\Gamma_D\binom{80}{0.8}$. Furthermore, the solutions from the set covering formulation with QoS requirements $\Gamma_D\binom{80}{1}$ and $\Gamma_D\binom{40}{0.6}$ use much fewer RSUs than the multi-flow ILP solution with QoS requirement $\Gamma_D\binom{80}{0.6}$.



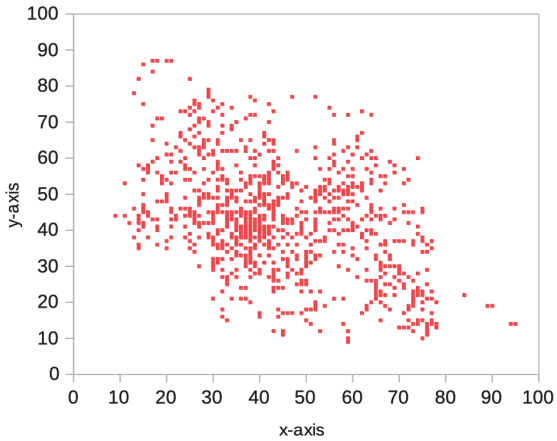
(a) Set covering ILP, $\rho = 0.6$, 247 RSUs



(b) Multi-flow ILP, $\rho = 0.6$, 693 RSUs

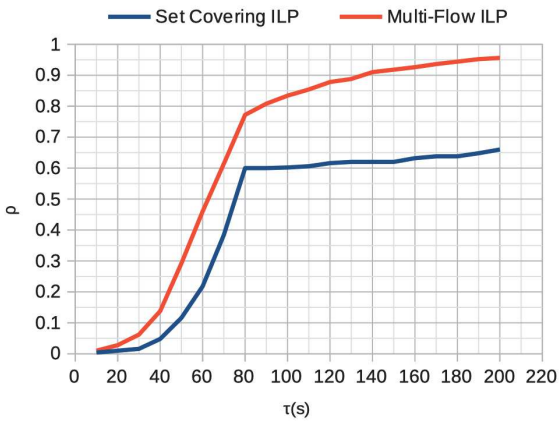


(c) Set covering ILP, $\rho = 0.8$, 359 RSUs

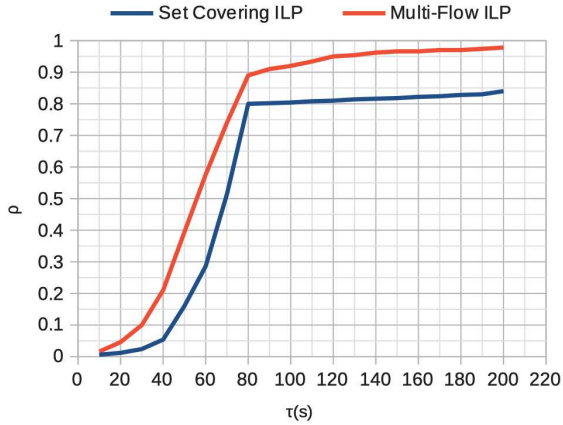


(d) Multi-flow ILP, $\rho = 0.8$, 807 RSUs

Figure 5.1: Layout of the deployments of RSUs obtained within a running time of 1 hour for $|T| = 500$ and $\tau = 80s$. Blue dots and red dots represent the RSUs deployed with the set covering and the multi-flow formulation, respectively.



(a) $\rho = 0.6$, red curve relates to deployment represented in Figure 5.1a and blue curve relates to deployment represented in Figure 5.1b



(b) $\rho = 0.8$, red curve relates to deployment represented in Figure 5.1c and blue curve relates to deployment represented in Figure 5.1d

Figure 5.2: Characteristic Γ_D curves for the deployments in Figure 5.1. The blue curves and the red curves represent the solutions obtained with the set covering formulation and the multi-flow formulation, respectively. All deployments have the parameters $|T| = 500$ and $\tau = 80s$.

Chapter 6

Conclusion

Vehicular networks are computer networks specifically designed to deal with cars and roadside units, providing many benefits to drivers, pedestrians and society. Gamma Deployment is a metric to evaluate the regularity of service delivered by a vehicular network. The Gamma Deployment Problem deals with the minimization of the number of roadside units ensuring the quality of service of the network under Gamma Deployment metric.

In this dissertation, we proved that the decision version of the Gamma Deployment Problem in Grids belongs to the complexity class NP-complete. This result is relevant since previous works proposed heuristics and integer linear programming formulation for the problem applied in grid graphs, even though there was no formal proof of NP-completeness.

Next, we proposed a correction for the multi-flow formulation present in the literature. This correction tackles the fact that the original formulation may obtain infeasible solutions for walks with cycles, whose existence in the dataset is implied by the discretization approach adopted. We also proposed a new integer linear programming formulation for the problem based on set covering and we proved that the polytope associated with its linear programming relaxation is contained in the polytope associated with the linear programming relaxation of the multi-flow formulation.

In computational experiments with a commercial solver, the set covering formulation widely outperforms the multi-flow formulation concerning the number of instances solved and linear programming relaxation gap. The experiments show an impact on the application of using the set covering formulation instead of the multi-flow formulation when the running time is limited: the costs decrease substantially since much fewer RSUs are necessary. Even though the deployments obtained with the set covering formulation use considerably fewer roadside units, both formulations provide the required

quality of service according to some Gamma Deployment curves. These curves also give an intuition of how a multi-objective optimization could be beneficial when there is a predetermined budget to invest in RSUs.

Interesting topics for future work include investigating the existence of parameterized-complexity and approximate algorithms for Gamma Deployment Problem and Gamma Deployment Problem in Grids. Besides, further complexity results for specific classes of graphs may reveal polynomial-time complexity in the domain of some of them. It would also be relevant to test the integer linear programming formulations with other datasets and compare their results with those provided by the heuristics proposed in the literature. Finally, a multi-objective optimization will be an appropriate approach to maximize the percentage of vehicles covered and minimize the inter-contact time when the budget to invest in RSUs is predetermined.

Bibliography

[1] AbdelSalam, H. S. and Olariu, S. (2009). Hexnet: Hexagon-based localization technique for wireless sensor networks. In *2009 IEEE International Conference on Pervasive Computing and Communications*, pages 1–6. ISSN .

[2] Aslam, B., Amjad, F., and Zou, C. C. (2012). Optimal roadside units placement in urban areas for vehicular networks. In *Computers and Communications (ISCC), 2012 IEEE Symposium on*, pages 423--429. IEEE.

[3] Barrachina, J., Garrido, P., Fogue, M., Martinez, F. J., Cano, J.-C., Calafate, C. T., and Manzoni, P. (2012). D-RSU: A Density-Based Approach for Road Side Unit Deployment in Urban Scenarios. In *International Workshop on IPv6-based Vehicular Networks (Vehi6), collocated with the 2012 IEEE Intelligent Vehicles Symposium*, pages 1–6. ISSN .

[4] Bekos, M. A., Gronemann, M., and Raftopoulou, C. N. (2016). Two-page book embeddings of 4-planar graphs. *Algorithmica*, 75(1):158--185.

[5] Capone, A., Cesana, M., Napoli, S., and Pollastro, A. (2007). Mobimesh: a complete solution for wireless mesh networking. In *2007 IEEE International Conference on Mobile Adhoc and Sensor Systems*, pages 1--3. IEEE.

[6] Eiza, M. H., Ni, Q., Owens, T., and Min, G. (2013). Investigation of routing reliability of vehicular ad hoc networks. *EURASIP Journal on Wireless Communications and Networking*, 2013(1):179. ISSN 1687-1499.

[7] Eze, E. C., Zhang, S., and Liu, E. (2014). Vehicular ad hoc networks (VANETs): Current state, challenges, potentials and way forward. In *Automation and Computing (ICAC), 2014 20th International Conference on*, pages 176--181. IEEE.

[8] Faraj, M. F., Sarubbi, J. F., Silva, C. M., and Martins, F. V. (2018). A memetic algorithm approach to deploy RSUs based on the gamma deployment metric. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, pages 1--8. IEEE.

[9] Faraj, M. F., Sarubbi, J. F. M., Silva, C. M., and Martins, F. V. C. (2017). A hybrid genetic algorithm for deploying RSUs in VANETs based on inter-contact time. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '17, pages 193--194, New York, NY, USA. ACM.

[10] Garey, M. R. and Johnson, D. S. (1977). The rectilinear steiner tree problem is np-complete. *SIAM Journal on Applied Mathematics*, 32(4):826--834.

[11] Heath, L. S. (1985). *Algorithms for embedding graphs in books.* PhD thesis, University of North Carolina at Chapel Hill.

[12] Itai, A., Papadimitriou, C. H., and Szwarcfiter, J. L. (1982). Hamilton paths in grid graphs. *SIAM Journal on Computing*, 11(4):676--686.

[13] Kchiche, A. and Kamoun, F. (2010). Centrality-based access-points deployment for vehicular networks. In *Telecommunications (ICT), 2010 IEEE 17th International Conference on*, pages 700--706. IEEE.

[14] Kumar, V., Mishra, S., and Chand, N. (2013). Applications of VANETs: Present & future. *Communications and Network*, 5:12--15.

[15] Liu, Y., Niu, J., Ma, J., and Wang, W. (2013). File downloading oriented roadside units deployment for vehicular networks. *Journal of Systems Architecture*, 59(10):938--946.

[16] Lochert, C., Scheuermann, B., Wewetzer, C., Luebke, A., and Mauve, M. (2008). Data aggregation and roadside unit placement for a vanet traffic information system. In *Proceedings of the fifth ACM international workshop on VehiculAr Inter-NETworking*, pages 58--65. ACM.

[17] Mershad, K., Artail, H., and Gerla, M. (2012). Roamer: Roadside units as message routers in VANETs. *Ad Hoc Networks*, 10(3):479--496.

[18] Owen, S. H. and Daskin, M. S. (1998). Strategic facility location: A review. *European journal of operational research*, 111(3):423--447.

[19] Owen, S. H. and Daskin, M. S. (2009). Facility location and supply chain management ? a review. *European Journal of Operational Research*, 196(2):401--412.

[20] Rashidi, M., Batros, I., Madsen, T. K., Riaz, M. T., and Paulin, T. (2012). Placement of road side units for floating car data collection in highway scenario. In *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on*, pages 114--118. IEEE.

[21] Reis, A. B., Sargento, S., and Tonguz, O. K. (2011). On the performance of sparse vehicular networks with road side units. In *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, pages 1--5. IEEE.

[22] Saini, M., Alelaiwi, A., and Saddik, A. E. (2015). How close are we to realizing a pragmatic VANET solution? a meta-survey. *ACM Computing Surveys (CSUR)*, 48(2):29.

[23] Sarubbi, J., Martins, F., and Silva, C. (2016). A genetic algorithm for deploying roadside units in VANETs. In *IEEE Congress on Evolutionary Computation (CEC'2016)*.

[24] Sarubbi, J. F. M. and Silva, C. M. (2016). Delta-r: A novel and more economic strategy for allocating the roadside infrastructure in vehicular networks with guaranteed levels of performance. In *Network Operations and Management Symposium (NOMS), 2016 IEEE/IFIP*, pages 665--671. IEEE.

[25] Silva, C. M., Guidoni, D. L., Souza, F. S., Pitangui, C. G., Sarubbi, J. F., Aquino, A. L., Meira, W., Nogueira, J. M. S., and Pitsillides, A. (2016a). Using the inter-contact time for planning the communication infrastructure in vehicular networks. In *Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on*, pages 2089--2094. IEEE.

[26] Silva, C. M., Guidoni, D. L., Souza, F. S., Pitangui, C. G., Sarubbi, J. F., and Pitsillides, A. (2016b). Gamma deployment: Designing the communication infrastructure in vehicular networks assuring guarantees on the v2i inter-contact time. In *Mobile Ad Hoc and Sensor Systems (MASS), 2016 IEEE 13th International Conference on*, pages 263--271. IEEE.

[27] Silva, C. M., Pitangui, C. G., Guidoni, D. L., Souza, F. S., and Sarubbi, J. F. (2016c). Deposição gamma: Alocando infraestrutura de comunicação para redes veiculares garantindo o intervalo entre contatos de veículos com a infraestrutura de comunicação. In *Simpósio Brasileiro de Pesquisa Operacional (SBPO), 2016*. SBPO.

[28] Silva, C. M. d. and Meira, W. (2015). Evaluating the performance of heterogeneous vehicular networks. In *Vehicular Technology Conference (VTC Fall), 2015 IEEE 82nd*, pages 1--5. IEEE.

[29] Sun, J. and Yang, Y. (2011). RSU localization model and simulation optimization for vii network. *Transport*, 26(4):394--402.

[30] Trullols, O., Fiore, M., Casetti, C., Chiasserini, C.-F., and Ordinas, J. B. (2010). Planning roadside infrastructure for information dissemination in intelligent transportation systems. *Computer Communications*, 33(4):432--442.

[31] Trullols-Cruces, O., Fiore, M., and Barcelo-Ordinas, J. (2012). Cooperative download in vehicular environments. *IEEE Transactions on Mobile Computing*, 11(4):663--678.

[32] Uppoor, S. and Fiore, M. (2011). Large-scale urban vehicular mobility for networking research. In *Vehicular Networking Conference (VNC), 2011 IEEE*, pages 62--69. IEEE.

[33] Xiong, Y., Ma, J., Wang, W., and Niu, J. (2012). Optimal roadside gateway deployment for VANETs. *Prz. ELEKTROTECHNICZNY*, (7):273--276.

[34] Zeadally, S., Hunt, R., Chen, Y.-S., Irwin, A., and Hassan, A. (2012). Vehicular ad hoc networks (VANETs): status, results, and challenges. *Telecommunication Systems*, 50(4):217--241.

[35] Zheng, Z., Sinha, P., and Kumar, S. (2009). Alpha coverage: Bounding the interconnection gap for vehicular internet access. In *INFOCOM 2009, IEEE*, pages 2831–2835. ISSN 0743-166X.