Marcelo Malta Rodrigues Martins

Strong Lower Bounds for the CVRP via Column and Cut Generation

DISSERTAÇÃO DE MESTRADO

DEPARTAMENTO DE INFORMÁTICA

Programa de Pós–Graduação em Informática

Rio de Janeiro
January 2016

Pontifícia Universidade Católica
do Rio de Janeiro

Marcelo Malta Rodrigues Martins

Strong Lower Bounds for the CVRP via Column and Cut Generation

DISSERTAÇÃO DE MESTRADO

Dissertation presented to the Programa de Pós–Graduação em Informática of the Departamento de Informática – PUC-Rio as partial fulfillment of the requirements for the degree of Mestrado.

Advisor: Prof. Marcus Vinicius Soledade Poggi de Aragão
Co-Advisor: Prof. Rafael Martinelli Pinto

Rio de Janeiro
January 2016

Pontifícia Universidade Católica
do Rio de Janeiro

Marcelo Malta Rodrigues Martins

Strong Lower Bounds for the CVRP via Column and Cut Generation

Dissertation presented to the Programa de Pós–Graduação em Informática of the Departamento de Informática of Centro Técnico Científico – PUC-Rio as partial fulfillment of the requirements for the degree of Mestrado.

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

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

Prof. Marco Serpa Mollinaro
Departamento de Informática – PUC-Rio

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

Prof. Marcio da Silveira Carvalho
Coordinator of the Centro Técnico Científico – PUC-Rio

Rio de Janeiro, January 18th, 2016

Marcelo Malta Rodrigues Martins

Marcelo Malta Rodrigues Martins obtained a BSc in Computer Science from Universidade do Estado do Rio de Janeiro (UERJ). During this course he held a scholarship from Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), maintaining an excellent academic performance. He has experience as a software developer and worked in OLX/BomNegocio and IBM.

Bibliographic Data

Malta, Marcelo

Strong Lower Bounds for the CVRP via Column and Cut Generation / Marcelo Malta Rodrigues Martins; Advisor: Marcus Vinicius Soledade Poggi de Aragão; Co–advisor: Rafael Martinelli Pinto. – 2016.

67 f: il. (color.) ; 30 cm

Dissertação de Mestrado – Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2016.

Inclui bibliografia

1. Informática – Dissertation. 2. Roteamento. 3. Geração de Colunas. 4. CVRP. 5. Pricing. 6. Cortes não robustos. I. Poggi, Marcus. II. Martinelli, Rafael. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. IV. Título.

CDD: 004

Acknowledgments

I would like to thank my dear wife Marianna, for being such a good life partner, and for understanding all we had to sacrifice to achieve this.

To my family that always believed in me.

To my advisor Marcus Poggi, for the opportunity and guidance through the path.

To my co-advisor Rafael Martinelli, for his involvement since the beginning. Always available to explain things, over and over again, for encouraging me through the dark times and for his belief.

To Pontifícia Universidade Católica do Rio de Janeiro, which provided a great environment to develop my work.

To CAPES, for the financial support and to the Brazilians who pay taxes.

To Professors Eduardo Uchoa and Marco Molinaro for the insightful inputs, which contributed to the improvement of this thesis.

Finally, to all my friends who contributed to make it easier, especially Ricardo, who always encouraged me.

Abstract

Malta, Marcelo; Poggi, Marcus; Martinelli, Rafael. **Strong Lower Bounds for the CVRP via Column and Cut Generation**. Rio de Janeiro, 2016. 67p. MSc Thesis – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

The *Capacitated Vehicle Routing Problem* (CVRP) is the seminal version of the vehicle routing problem, a classical problem in Operational Research. Introduced by Dantzig e Ramser, the CVRP generalizes the Traveling Salesman Problem (TSP) and the Bin Packing Problem (BPP). In addition, routing problems arise in several real world applications, often in the context of reducing costs, polluent emissions or energy within transportation activities. In fact, the cost with transportation can be roughly estimated to represent 5% to 20% of the overall cost of a delivered product. This means that any saving in routing can be much relevant. The CVRP is stated as follows: given a set of $n + 1$ locations – a depot and n customers – the distances between every pair of locations, integer demands associated with each customer, and a vehicle capacity, we are interested in determining the set of routes that start at the depot, visits each customer exactly once and returns to the depot. The total distance traveled by the routes should be minimized and the sum of the demands of customers on each route should not exceed the vehicle capacity. This work considers that the number of available vehicles is given. State of the art algorithms for finding and proving optimal solutions for the CVRP compute their lower bounds through column generation and improving it by adding cutting planes. The columns generated may be elementary routes, where customers are visited only once, or relaxations such as q-routes and the more recent ng-routes, which differ on how they allow repeating customers along the routes. Cuts may be classified as robust, those that are defined over arc variables, and non-robust (or strong), those that are defined over the column generation master problem variables. The term robust used above refers to how adding the cut modifies the efficiency of solving the pricing problem. Besides the description above, the most efficient exact algorithms for the CVRP use too many elements turning its replication a hard long task. The objective of this work is to determine how good can be lower bounds computed by a column generation algorithm on ng-routes using only capacity cuts and a family of strong cuts, the limited memory subset row cuts. We assess the leverage achieved with the consideration of this kind of strong cuts and its combination with others techniques like Decremental Space State Relaxation (DSSR), Completion Bounds, ng-Routes and Capacity Cuts over a Set Partitioning formulation of the problem. Extensive computational experiments are presented along with an analysis of the results obtained.

Keywords

Routing; Column Generation; CVRP; Pricing; Non-robust cuts;

Resumo

Malta, Marcelo; Poggi, Marcus; Martinelli, Rafael. **Limites Inferiores Fortes para o CVRP via Geração de Colunas e Cortes**. Rio de Janeiro, 2016. 67p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

O *Capacitated Vehicle Routing Problems* (CVRP) é uma versão seminal do problema de roteamento de veículos, um clássico problema em Pesquisa Operacional. Introduzido por Dantzig e Ramser, o CVRP generaliza o Traveling Salesman Problem (TSP) e o Bin Packing Problem (BPP). Problemas de roteamento aparecem em diversas aplicações no mundo real, geralmente no contexto de diminuição de custos, emissão de poluentes ou energia dentro das atividades relacionadas ao transporte. De fato, estes custos podem ficar entre 5% e 20% do custo total do produto. Por isto, qualquer economia nos custos de roteamento pode ser relevante. O CVRP é definido da seguinte maneira: dado um conjunto de $n + 1$ localidades – um depósito e n clientes – as distâncias entre cada par de localidades, as demandas inteiras associadas a cada cliente e a capacidade do veículo, quer se obter um conjunto de rotas que comecem no depósito, visitem cada cliente apenas uma vez e retornem ao depósito. A distâncias percorrida deve ser mínima e a soma das demandas dos clientes presentes em cada rota não pode exceder a capacidade do veículo. Este trabalho considera que o número de veículos disponíveis é conhecido. Algoritmos no estado da arte para encontrar e provar que uma solução é ótima, para o CVRP, calculam seus limites inferiores através de geração de colunas e depois os melhoram com a adição de planos de corte. As colunas geradas podem ser rotas elementares, onde obrigatoriamente cada cliente é visitado somente uma vez, ou uma relaxação desta obrigação com o uso de q-rotas ou ng-rotas, que diferem apenas em como é permitido que um cliente seja revisitado dentro de uma mesma rota. Já os cortes são classificados como robustos, aquele que são definidos sobre as variáveis dos arcos, e não robustos (ou fortes), que são os definidos sobre as variáveis do problema mestre da geração de colunas. O termo robusto, usado acima, se refere a como a adição do corte modifica a eficiência da resolução do problema de pricing. Além do descrito acima, o algoritmo exato mais eficiente para o CVRP usa muitos elementos, o que torna sua replicação uma tarefa difícil e longa. O objetivo deste trabalho é determinar o quão bom são os limites inferiores obtidos com geração de colunas de ng-rotas usando apenas cortes de capacidade e os recentes subset row cuts de memória limitada. Além disto, é avaliado o ganho conseguido com a consideração deste tipo de corte forte e as combinações com outras técnicas, como por exemplo, Decremental Space State Relaxation (DSSR), Completion Bounds, ng-rotas e cortes de capacidade sobre a formulação de Set Partitioning. Extensos experimentos computacionais são apresentados em conjunto com a análise dos resultados obtidos.

Palavras–chave

Roteamento; Geração de Colunas; CVRP; Pricing; Cortes não robustos;

Contents

List of Figures

List of Tables

I
Introduction

Since the first appearance in the literature in the late 50's in the original work of Dantzig and Ramser [1], the Capacitated Vehicle Routing Problem (CVRP) has been widely studied by many researchers. Many contributions have been made in several directions since then. Some researchers focus on finding an exact algorithm, while others are concentrating efforts to find quasi-optimum solutions, meta-heuristics and hybrid methods. The objective of this text is to study the most recent proposed exact methods and understand the leverage gained in order to propose simple algorithms to find high quality lower bounds.

As shown in Toth and Vigo [2], it is estimated that the cost with the transportation can roughly be between 5% to 20% of the overall cost of a delivered product. This means that any saving in routing can be much relevant. Here is where the routing problem plays a significant role in real life scenario.

The CVRP can be defined as follows: a set of customers, each with a demand, needs to be serviced by a number of vehicles starting and ending at a central depot. Each customer needs to be visited exactly once while capacity of vehicle must not be exceeded. The objective is to service all customers traveling the minimum distance.

The CVRP is classified as being an $\mathcal{NP}$-hard problem, therefore the use of exact optimization methods or simple enumerations of all feasible solutions may be difficult in acceptable CPU times, especially when the problem involves real-world data sets or when the instances are very large. Furthermore, it is easy to prove its $\mathcal{NP}$-completeness, since a CVRP with only one vehicle is the *Traveling Salesman Problem* (TSP) itself. As a result, there is no known polynomial time algorithm to find and prove its solution to be optimal. Exact methods to solve this problem rely on the use of lower bounds. The lower bounds proposed in the last thirty years are almost always based on mixed integer programming formulations.

Many formulations were suggested for this problem in the literature. Maybe the most basic one is the formulation based on a multi-commodity flow, introduced by Ford et al. [3]. Although this formulation has a polynomial number of constraints and variables, its linear relaxation is weak and, because of that, the efficiency of the method turned out to be quite limited. Another example of formulation is the one suggested by Laporte [4], the *Two-Index Formulation* (TIF), which has a stronger linear relaxation and successfully solved small instances using *Branch-and-Cut* (BC). Until nowadays this is usually the best formulation to solve instances where the

routes in the solution must service many customers. It should be noted that this formulation has an exponential number of constraints, what is most often not an issue when computing the corresponding lower bound. A third formulation, which is central to this work is the *Set Partitioning Formulation* (SP) [5]. Usually, this formulation's linear relaxation produces a strong lower bound. The most successful recent exact algorithms use this formulation. The drawback of the SP formulation is that it has an exponential number of variables. Therefore it is necessary to use a *Column Generation* approach in order to deal with this large number of variables.

Column Generation is a technique used to solve formulations with an exponential number of variables. On each iteration, just a subset of variables is considered and, based on the current solution, one or more new variables with minimum reduced cost are generated through the resolution of a combinatorial optimization problem. Thus, the problem is solved iteratively, and the selection (pricing) of the new variables to be inserted in the formulation leads to improvements on the solution, i.e. they must have negative reduced cost. This auxiliary combinatorial optimization problem mentioned above is then called the pricing sub-problem.

CVRP formulations are limited to consider only routes that visit each customer exactly once. These routes are called elementary routes. When just such columns are considered, the column generation sub-problem correspond to solve the *Elementary Shortest Path Problem with Resource Constraint* (ESPPRC), which is known to be strongly $\mathcal{NP}$-hard as shown by Dror [6]. The hardness of ESPPRC has motivated Christofides, Mingozzi and Toth [7] to also consider columns that are not elementary. They proposed the so-called q-routes, where customers can be visited as many times as desired as long as the demand of each customer is counted every time it is visited. The column generation sub-problem now corresponds to solve the *Shortest Path Problem with Resource Constraints* (SPPRC), which is weakly $\mathcal{NP}$-hard, i.e., there are pseudo-polynomial algorithms available for solving it.

Naturally, relaxing the routes implies having weaker lower bounds. Tighter lower bounds can be obtained by enforcing partial elementarity on the set of q-routes. A first effort is to forbid the q-routes to have cycles of small sizes. The larger is the size of the smallest cycle in the q-routes the better is the lower bounds obtained. Unfortunately, the complexity of eliminating cycles grows with the factorial of the size of the largest eliminated cycle, shown by Irnich and Villeneuve [8]. This motivated the proposition of another route relaxation: ng-routes, introduced by Baldacci et al. [9]. It intends to prevent cycles among customers in a given neighborhood.

The CVRP lower bounds obtained with the SP formulation can be strengthened by adding polyhedral cuts. This work considers only two classes of cuts: (i) The Rounded Capacity Cuts, which are defined over the variables of the TIF formulation and can be directly translated to the variables of the SP formulation. (ii) The Limited Memory Subset Row Cuts, which are a generalization of the Subset Row Cuts and are defined over the variables of the SP formulation. This

dissertation aims at developing pricing and separation algorithms for finding lower bounds for the CVRP using the SP formulation with ng-routes and these two classes.

I.1 Literature Review

As briefly mentioned before, the first algorithm to solve the SPPRC was proposed by Christofides et al. citechristofides-qroute. It introduced the idea of q-routes without 2-cycles, which allows multiple visits to a customer in a same route, on the condition that at least two other customers are visited between successive visits. The elimination of 2-cycles does not change the overall complexity of the pricing algorithm, but it improves significantly the lower bounds obtained. Later, Irnich and Villeneuve [8] generalized q-routes to eliminate k-cycles, but this included a factor of $k!$ to the time complexity of the pricing. Fukasawa et al. [10] showed that is not worthy to use $k > 4$. Besides, they were able to solve to optimality all instances in the literature with up 135 customers using a Branch-Cut-and-Price algorithm using only cuts on the Two-Index Formulation.

Later in 2008, Baldacci et al. [11] solved almost all instances in Fukasawa et al. [10] work within considerably less computational time. They were able to obtain good estimates for the optimal dual variables values, leading the algorithm to converge in fewer iterations. Moreover, the cuts on the Two-Index Formulation and other cuts for the Set Partitioning formulation as Strengthened Capacity Cuts and Clique Cuts were also used. In the same year, Pessoa et al. [12] improved upon Fukasawa et al.[10] using cuts from an extended formulation with capacity index. Differently from the cuts on the Set Partitioning Formulation, these cuts do not change the complexity of the dynamic programming algorithm used to price the q-routes.

Three years after their previous paper, Baldacci et al. [9] introduced a new route relaxation named ng-routes. Those routes prevent cycles within the neighborhood of each customer. This new relaxation improves a lot the bounds obtained with q-routes. Using this new relaxation along with Subset Row Cuts and Weak Subset Row Cuts, the results presented in Baldacci et al. [11] were improved, but no new optimal solution was proved.

In 2011, Martinelli et al. [13] were able to increase the size of the neighborhoods of customers which are forbidden. The performance of the exact pricing was enhanced by using Decremental State Space Relaxation (DSSR), proposed by Righini and Salani [14], and completion bounds. This technique iteratively grows the size of the neighborhoods on which cycles are forbidden. This idea was later extended by Martinelli et al. [15] leading to a speed up of the solution of the ESPPRC.

In 2012, Contardo [16] successfully solved for the very first time the hard instance M-n151-k12. To obtain this result, he combined cuts from the Two-Index Formulation with the Strong Strengthened Capacity Cuts and the Subset Row Cuts. The columns were q-routes without 2-cycles. Although this leads to a poor

relaxation, the partial elementarity of the routes was enforced by the Strong Degree Cuts. These cuts are an alternative way to prevent a customer to be revisited and consequently guaranteeing partial elementarity. In the sequence, Røpke [17] also solved successfully the instance M-n151-k12 after a long run of a Branch-Cut-and-Price algorithm with specialized branching rules and strong branching – it took almost 6 days.

In 2014, Contardo and Martinelli [18] combined their previous works ([13], [16]). Using ng-route relaxation, DSSR and completion bounds, the pricing was improved. Strengthened Capacity Cuts, Subset Row Cuts and Strong Degree Cuts were also considered. With these changes, they were able to solve for the very first time the hard instance F-n135-k7 using only a column generation approach.

And finally, also in 2014, Pecin et al. [19] introducted the Limited Memory Subset Row Cuts. They are a weakening of the traditional Subset Row Cuts, introduced in Jepsen et al. [20], which are Chvátal-Gomory rank-1 cuts. This weakening allows the pricing to be dynamically adjusted, making it much less costly and yet without compromising its effectiveness. Combining this relaxation with elements from all previous work the algorithm was able to solve all the classical instances from the literature with up to 200 vertices in reasonable times, including the hard instances M-n200-k17 and M-n200-k16.

I.2 Motivation

The algorithm proposed by Pecin et al. [21] was the first one to solve all the classical instances from the literature with up to 200 vertices. However, this algorithm was a combination of several methods and it is not clear what was the improvement made by each one. It is believed that lm-SRC was the main contribution for this improvement.

Therefore, the objective of this work is to devise a pure implementation of the algorithm proposed in Pecin et al. [21], using only the lm-SRC combined with Rounded Capacity Cuts on the SP formulation considering ng-routes. This has not been done yet and will enlighten the improvement that can be obtained on the lower bounds. To evaluate this approach, we run the algorithm for the same instances that Pecin et al. [21] and we also present the lower bounds for different sizes of ng-sets.

I.3 Thesis Outline

This thesis is structured as follows.

– Chapter II presents the two most used formulations to solve the CVRP.

– Chapter III explains how to use a Column Generation approach to solve the Set Partitioning Formulation. It also shows techniques to improve the performance of pricing new columns.

– Chapter IV introduces examples of cuts that can be separated in attempt of improving the bounds obtained.

– Chapter V presents computational results.

– Chapter VI draws some insights from what is reported and suggest some future work.

II
Mathematical Formulations

This section presents two formulations to solve the CVRP. We start with the Two Index Formulation (TIF), which was proposed by Laporte and Nobert [4]. Then, we present the Set Partitioning Formulation proposed by Balinski and Quandt [5], which is the widely adopted formulation to solve the CVRP.

II.1 Two Index Formulation

The CVRP considers a complete and undirected graph $G = (V, E)$ where $V = \{0, \ldots, n\}$ represents the vertex set and E the edges set. Vertices in $N = \{1, \ldots, n\}$ correspond to the customers, whereas the depot is represented by 0. Each edge $(i, j) \in E$ can be represented by a single index $e = (i, j)$ and it has a nonnegative cost c_{ij} to be traversed and x_e denotes the number of times the edge is traversed by a vehicle. Each customer i has a nonnegative demand q_i and Q represents the capacity of each vehicle k available in the fleet K. Given a set of customers $S \subseteq V$, let $\delta(S)$ denote the set of edges $e \in E$ which have only one endpoint in S. Let $q(S) = \sum_{i \in S} q_i$ and let $r(S)$ be a lower bound on the number of vehicles needed to service the customers on set S. The overall objective is to minimize the total cost of the edges traversed. The Two Index Formulation is then defined as follows:

$$\text{(TIF)} \qquad \min \quad \sum_{e \in E} c_e x_e \tag{II.1}$$

subject to

$$\sum_{e \in \delta(\{i\})} x_e = 2 \qquad \forall i \in N \tag{II.2}$$

$$\sum_{e \in \delta(\{0\})} x_e = 2|K| \tag{II.3}$$

$$\sum_{e \in \delta(S)} x_e \geq 2r(S) \qquad \forall S \subseteq N \tag{II.4}$$

$$x_e \in \{0, 1\}, \qquad \forall e \in E \setminus \delta(0) \tag{II.5}$$

$$x_e \in \{0, 1, 2\} \qquad \forall e \in \delta(0) \tag{II.6}$$

Constraints (II.2) guarantee that each customer is serviced by just one vehicle (it must arrive and leave the customer). In addition, the constraint (II.3) imposes

that if we take the set of edges with an endpoint at the depot, considering that we have $|K|$ vehicles leaving and returning to it, then there are $2|K|$ edges from this set being traversed. These two constraints are also known as the *degree constraints* and they are part of the polynomial-sized constraints of the formulation. The constraints (II.4) are the *capacity constraints* which, for any subset S of customers, state that at least $r(S)$ vehicles must enter and leave S. Note that these constraints, besides assuring that no vehicle will have its capacity violated, they also eliminate subtours. The exact way to obtain $r(S)$ is by solving a Bin Packing problem, which is known to be an $\mathcal{NP}$-Hard problem [22]. However, the formulation remains valid if $r(S)$ is replaced by a lower bound on its value, such as $k(S) = \lceil q(S)/Q \rceil$. Constraints (II.5) impose that x_e can be 1 or 0, representing whether the edge was traversed or not. And finally, the constraints (II.6) allow some x_e to be also 2 if the edge has an endpoint at the depot, allowing routes with only one customer to be build. Those last two constraints are the *integrality constraints*.

Until today this is the best formulation to solve instances in which there are routes with too many customers to be service on the solutions found. It should be noted that this formulation has an exponential number of constraints, what is often not an issue when computing its corresponding lower bounds.

II.2 Set Partitioning

The Two Index Formulation can be rewritten using a variable for each feasible route. A route is a path which starts at the depot, traverses a sequence of customers and finishes back to depot without violating the vehicle's capacity Q. Let Ω represent the set of all feasible routes. We define a binary variable λ_r which represents whether the route $r \in \Omega$ is used or not. Let b_r^e be a constant value indicating how many times route r traverses edge e. The reformulation is as follows.

$$\min \quad \sum_{e \in E} c_e x_e \tag{II.7}$$

subject to

$$\sum_{r \in \Omega} b_r^e \lambda_r = x_e \qquad \forall e \in E \tag{II.8}$$

$$\sum_{e \in \delta(\{i\})} x_e = 2 \qquad \forall i \in N \tag{II.9}$$

$$\sum_{e \in \delta(\{0\})} x_e = 2|K| \tag{II.10}$$

$$\lambda_r \in \{0,1\} \qquad \forall r \in \Omega \tag{II.11}$$

$$x_e \in \{0,1\} \qquad \forall e \in E \setminus \delta(0) \tag{II.12}$$

$$x_e \in \{0,1,2\} \qquad \forall e \in \delta(0) \tag{II.13}$$

Constraints (II.8) bind varaibles x_e and λ_r. x_e is given by the number of times

an edge is traversed by all routes in the set Ω. Now we can replace variables x_e using this relation with λ_r to obtain a new formulation.

$$\min \qquad \sum_{e \in E} c_e \sum_{r \in \Omega} b_r^e \lambda_r \tag{II.14}$$

subject to

$$\sum_{e \in \delta(\{i\})} \sum_{r \in \Omega} b_r^e \lambda_r = 2, \qquad \forall i \in N \tag{II.15}$$

$$\sum_{e \in \delta(\{0\})} \sum_{r \in \Omega} b_r^e \lambda_r = 2|K| \tag{II.16}$$

$$\lambda_r \in \{0, 1\} \qquad \forall r \in \Omega \tag{II.17}$$

Let $a_r^i \in \{0, 1\}$ denote if a customer i is visited by a route r. Knowing that a route traverses two adjacent edges when servicing a customer and when leaving and returning to the depot, we can write $\sum_{e \in \delta(\{i\})} b_r^e = 2a_r^i, \forall i \in V$ and $\forall r \in \Omega$. Analogously, we can also write the cost of a route $c_r = \sum_{e \in E} c_e b_r^e, \forall r \in \Omega$. Finally, we can write the Set Partitioning Formulation as follows.

$$(SP) \qquad \min \qquad \sum_{r \in \Omega} c_r \lambda_r \tag{II.18}$$

subject to

$$\sum_{r \in \Omega} a_{ir} \lambda_r = 1, \qquad \forall i \in N \tag{II.19}$$

$$\sum_{r \in \Omega} \lambda_r = |K|, \tag{II.20}$$

$$\lambda_r \in \{0, 1\} \qquad \forall r \in \Omega \tag{II.21}$$

Constraints (II.19) impose that a customer i is serviced by only one route $r \in \Omega$. Constraint (II.20) assures that $|K|$ vehicles are used. And finally, constraints (II.21) force the variables λ_r to be binary.

Despite the fact that at first the formulation seems quite simple, the set Ω can contain an exponential number of routes which results in an exponential number of variables. So, to handle that it is necessary the use of a *Column Generation* approach, as presented in Chapter III.

It is important to notice that it is not necessary that set Ω contain only elementary routes because the formulation will discard any route with coefficient $a_{ir} > 1$ through the constraint (II.19), since the λ_r variables are binary. This behavior makes the task to solve the Column Generation easier, but the quality of the lower bounds obtained may drop significantly, as we will discuss in Chapter III.

III
Column Generation

Column generation is a technique that allows dealing with linear programs with a large number of variables, usually the ones on which the number of variables grows exponentially with the size of the problem. This can be exploited when the search for a variable with smallest reduced cost can be done by solving a combinatorial optimization problem. The overall idea is to split the linear programming problem in two problems: the restricted master problem and the pricing sub-problem. Analogous to the revised simplex method, the restricted master considers only a small subset of the variables. In order to obtain new columns/variables to improve the restricted master solution, an auxiliary problem called pricing sub-problem is solved. The input for the pricing problem are the values of the dual variables at the current restricted master optimal solution. Determining whether there are new columns with potential to improve the master solution reduces to check whether the optimum solution value for the pricing sub-problem is strictly negative. Columns with negative reduced cost are added to the restricted master which is then solved again. This iterative process continues until no suitable column is priced from the sub-problem.

The Set Partitioning Formulation, presented in Section II.2, requires enumerating all possible routes which a vehicle can perform. The number of routes grows exponentially with the number of customers. When only routes that visit exactly once each customer are considered, the pricing sub-problem corresponds to the Elementary Shortest Path Problem with Resource Constraints (ESPPRC), where the resource constraints refer to the use of the capacity of the vehicle regarding the demand of each customer visited along the route. To avoid the task of solving the strongly $\mathcal{NP}$-hard ESPPRC problem at each iteration, a relaxation of the route definition may be considered, including in the SP formulation columns which correspond to routes that visit customers more than once. The pricing problem is then the Shortest Path Problem with Resource Constraints (SPPRC). Allowing these multiple visits to customers may weaken significantly the lower bounds obtained on the linear relaxation of the SP formulation. A way of strengthening the lower bounds would be to propose forms of controlling the number of additional visits to each customer in a route. There are two main approaches in the literature to achieve this objective. The first one aims at eliminating cycles of a given size along the relaxed routes. The second comes from observing that cycles are more likely to occur among customers that are close to each other. As a consequence it explores

this idea by forbidding the relaxed route to return to a customer that belongs to a neighborhood set of other customers that are close to it. Next section provides more details of the pricing problem and how the relaxation works, allowing multiple visits along a route. The following sections are dedicated to the approach that exploits the idea of avoiding many visits to customers. It reduces to find minimum reduced cost ng-routes, as these routes are called.

III.1 Pricing

The reduced cost of a column is given by Equation (III.1) where π_i denote the dual variables associated to constraints (II.19) and π_0 is the dual variable associated to constraint (II.20). Thus, the reduced cost of a route is its original cost c_r minus the dual variable π_0 and the dual variable π_i for each customer i visited by this route.

$$\overline{c}_r = c_r - \pi_0 - \sum_{i \in N} a_{ir}\pi_i \tag{III.1}$$

The dual variables associated with each customer can be split equally between the edge entering and the one leaving the customer. Figure III.1 illustrates this idea. Therefore, the reduced cost can be rewritten as a function of edges as shown in Equation (III.2).

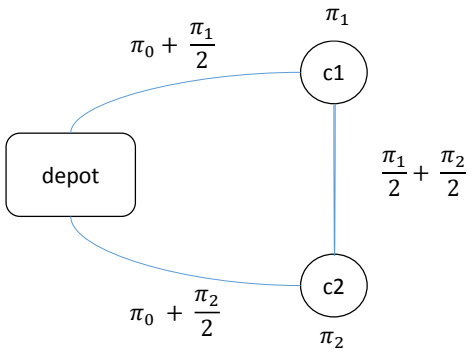

Figure III.1: Edge's reduced cost associated to customers' dual variables

$$\overline{c}_e = c_e - (\pi_u + \pi_v)/2 \tag{III.2}$$

At start the column generation procedure needs to construct an initial feasible subset of variables, so that the restricted master can be first solved. The initial set Ω can be constructed taking one route at time and greedily visiting customer which has not been visited yet, without exceeding the vehicle capacity. This heuristic can yield more than $|K|$ routes, however in our experiments it did not occur when using the classical instances for the problem. The pseudo code to the procedure is shown in Algorithm III.1.

The general pricing algorithm builds a dynamic programming matrix $\mathcal{M}$ of size $(Q+1) \times |N|$, where each entry $\mathcal{M}(q, i)$ contains the list of all paths which

Algorithm III.1 Algorithm to generate initial Ω set

```
1: procedure BUILDOMEGA
2:     demand ← 0
3:     routes ← |K|
4:     Visited ← ∅
5:     Ω ← {}
6:     while routes ≥ 0 do
7:         newRoute ← {}
8:         newRoute ← newRoute ∪ {0}
9:         for v ∈ N do
10:            if v ∉ Visited  and  (demand + q_v ≤ Q) then
11:                demand ← demand + q_v
12:                Visited ∪ {v}
13:                newRoute ∪ {v}
14:        newRoute ← newRoute ∪ {0}
15:        route ← route ∪ {newRoute}
16:        routes ← routes − 1
```

starts at the depot vertex, visits a set of customers and ends at customer i with cumulative demand q. At the beginning of the algorithm, the matrix is empty except for entry $\mathcal{M}(0,0)$ which contains one single element representing the path which has not yet left the depot. The algorithm then fills the matrix iteratively for each customer from cumulative demand 1 up to Q. On each step, when analyzing entry $\mathcal{M}(q,i)$, all possible extensions to customer i and cumulative demand q are considered and stored in the matrix. The process terminates when the whole matrix is filled and the algorithm returns any route with negative reduced cost found on entries $\mathcal{M}(q,0), q \in \{1,\ldots,Q\}$.

Associated to each path in the matrix, the algorithm stores a label containing a set of required information. For the general pricing algorithm, the label associated to a path P is defined as $\mathcal{L}(P) = (v(P), q(P), N(P), \bar{c}(P))$, where $v(P)$ is the last customer of the path, $q(P)$ is the total demand, $N(P)$ is the set of visited customers, and $\bar{c}(P)$ is the total reduced cost.

Note that the dynamic programming algorithm stores all possible paths from the depot to each customer without violating the available capacity. Clearly, this approach is exponential and may be prohibitive for all but some small instances. The first effort to reduce the number of labels generated by the algorithm is to use a dominance rule, which discards a dominated label known to provide a reduced cost no better than a dominant one. Given two labels $\mathcal{L}(P_1)$ and $\mathcal{L}(P_2)$, $\mathcal{L}(P_1)$ dominates $\mathcal{L}(P_2)$ if the following conditions hold.

(i) $v(P_1) = v(P_2)$

(ii) $q(P_1) \leq q(P_2)$

(iii) $N(P_1) \subseteq N(P_2)$

(iv) $\bar{c}(P_1) \leq \bar{c}(P_2)$

The first three conditions guarantee that path P_1 is allowed to perform at least the same possible paths back to the depot as P_2 and the last one guarantees that any of these completions back to the depot results in a better reduced cost when taken from P_1 than from P_2. This concludes that $\mathcal{L}(P_2)$ is dominated by $\mathcal{L}(P_1)$.

The goal of the pricing problem presented is to find one or more elementary routes which do not exceed the vehicle capacity and have negative reduced cost. As mentioned before, this is an $\mathcal{NP}$-hard problem and for this reason we chose to solve the SPPRC, for which the time complexity to fill the matrix is $\mathcal{O}(|N|^2 Q)$. The idea is to keep just the best path on each entry $\mathcal{M}(q, i)$. This can be done by using the recurrence presented in Equation (III.3).

$$\begin{cases} T(q_i, i) = & \bar{c}_{0i} \\ T(D, i) = & \min\limits_{j \in N} T(D - q_i, j) + \bar{c}_{ji}, \end{cases} \tag{III.3}$$

A first approach to generated relaxed routes was introduced by Christofides, Mingozzi and Toth [7]. They proposed the so called q-routes. In their work, They showed that restricting q-routes to forbid 2-cycles does not change the pricing complexity and it improves the bounds for the SP formulation. Figure III.2 illustrates the generalization of q-routes without k-cycles, where k stands for the length of a cycle.

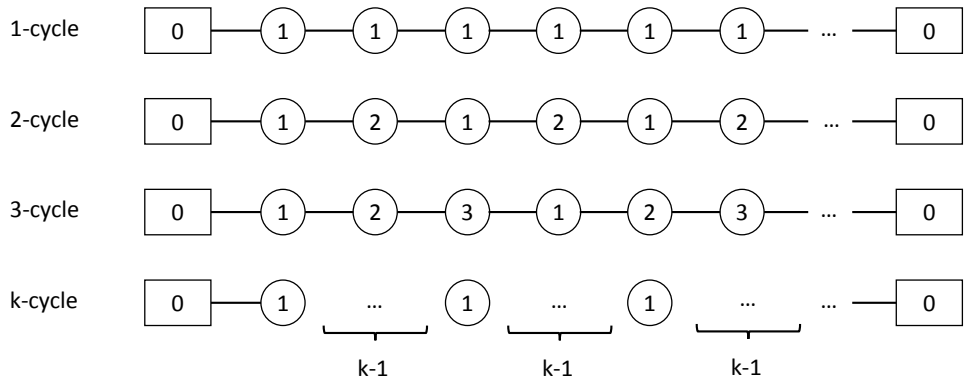

Figure III.2: Examples of routes with k-cycles

It is clear that larger values for k improve the lower bounds for the SP formulation. However, Fukasawa et al. [10] showed that it is not worthy to consider $k > 4$, due to its impact over pricing complexity. In fact, k-cycle elimination impacts the pricing complexity by a factor of $k!$. Unfortunately, 4-cycle elimination is still not enough to obtain lower bounds near the ones obtained only considering elementary routes.

III.2 Forbidding return to close customer neighbors: ng-Routes

Recently, Baldacci et al. [9] introduced a new approach, called the *ng*-route relaxation. The *ng*-route relaxation also allows a customer to be visited more than once but instead avoiding cycles of fixed length, it avoids cycles based on memory sets $NG_v \subseteq N$, associated with each customer v. Knowing that cycles are more likely to appear in the neighborhood of a given customer v, the memory sets are usually built using the closest neighbors of each customer. The size of the *ng*-sets is represented by $\Delta(NG)$ and it is defined *a priori*. Although larger values for $\Delta(NG)$ yield routes closer to elementarity, they also will impact in the overall complexity of the pricing.

Let $\Pi(P)$ be the customer set replacing the set $N(P)$ of visited customers in the labels used by the dynamic programming algorithm. Its meaning is changed to be the set of visited customers path P *remembers* so far. Thus, a label can now be defined as $\mathcal{L}(P) = (v(P), q(P), \Pi(P), \bar{c}(P))$. An extension from label $\mathcal{L}(P)$ to a customer v_{p+1} can only occurs, if and only if $v_{p+1} \notin \Pi(P)$ and $q(P)+q_{v_{p+1}} \leq Q$. After a successful extension, a new label $L(P')$ is created for path $P' = (0, \ldots, v_p, v_{p+1})$ by the operation presented in Equation (III.4).

$$\mathcal{L}(P') = (v_{p+1}, q(P) + q_{v_{p+1}}, \Pi(P) \cap N_{v_{p+1}} \cup \{v_{p+1}\}, \bar{c}(P) + \bar{c}_{v_p v_{p+1}}) \tag{III.4}$$

Clearly with the introduction of $\Pi(P)$, the dominance rule changes. Condition (iii) presented in Section III.1 now turns into $\Pi(P_1) \subseteq \Pi(P_2)$, meaning that label $\mathcal{L}(P_1)$ dominates label $\mathcal{L}(P_2)$ only when its memory is a subset of $\mathcal{L}(P_2)$ memory, i.e. any allowed extension from path P_2 can also be made from path P_1. This clearly reinforces the dominance rule based on the fact that, for any path P, $\Pi(P) \subseteq N(P)$, leading to a reduction on the number of labels.

As showed in Equation (III.4), the memory set $\Pi(P)$ is updated at each extension with the intersection of all *ng*-sets of all customers previously visited, united by the current customer. The overall contruction of the memory set is presented in Equation (III.5). Notice that it is easy to see $|\Pi(P)| \leq |C(P)|$, but even with this reduction, the size of $\Pi(P)$ still has an impact in the pricing complexity.

$$\Pi(P) = \left\{ i_k \in C(P) : i_k \in \bigcap_{s=k+1}^{P} NG_s, k = 1, \ldots, p-1 \right\} \cup i_p \tag{III.5}$$

To make the extension process clearer, we present two figures. In Figure III.3, by the time the extension from customer 1 to 2, 1 is forgotten (since $1 \notin NG_2$) and it is available to be revisited immediately or in the next extensions. Figure III.4 presents the opposite situation. Since $3 \in NG_2$, 3 is not allowed to be revisited, which forbids the extension.

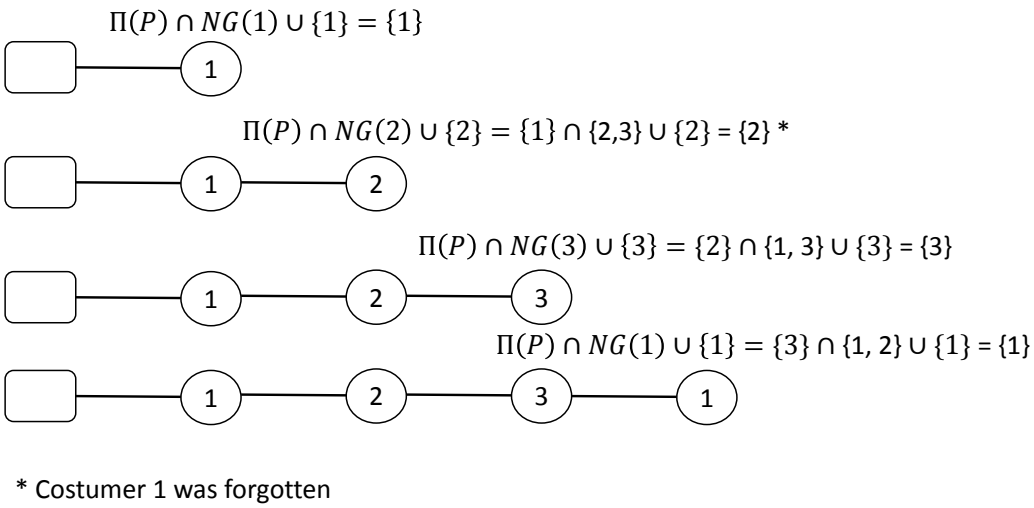

Figure III.3: Example of an allowed extension. $NG(1) = \{1, 2\}, NG(2) = \{2, 3\}, NG(3) = \{1, 3\}$

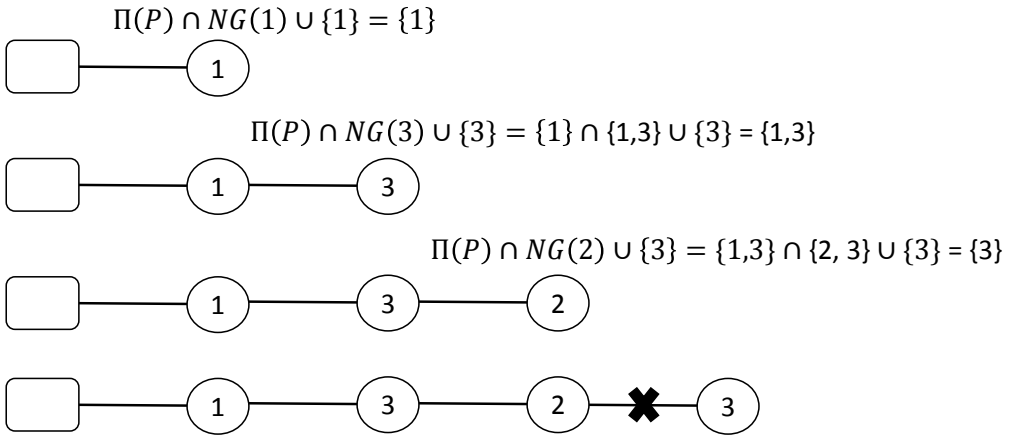

Figure III.4: Example of a forbidden extension. $NG(1) = \{1, 2\}, NG(2) = \{2, 3\}, NG(3) = \{1, 3\}$

If a given extension is to the depot, then we successfully build an *ng*-route. And from this point, there is no other extensions available. One problem arises with solving the SPPRC with this route relaxation, the exponential number of labels that can be generated and stored. The mitigation of this "curse of dimensionality" is the main concern when build an algorithm to solve the problem. The control mechanism and the algorithm to solve the problem will be formally introduced in the next section.

As mentioned previously, the *ng*-route relaxation improves the pricing problem by reducing the number of labels stored because of the changes done in the dominance rule. A label $\mathcal{L}(P_1)$ dominates another label $\mathcal{L}(P_2)$ if and only if, every feasible extensions from P_2 can be done with better or equal reduced cost when taken from P_1. Therefore P_2 can be removed from the list of labels. It is interesting

to check whether a label dominates or is dominated, whenever this label is created and the label list should be ordered by reduced cost for simplicity. Thus, the new sufficient conditions are:

(i) $v(P_1) = v(P_2)$

(ii) $q(P_1) \leq q(P_2)$

(iii) $\Pi(P_1) \subseteq \Pi(P_2)$

(iv) $\overline{c}(P_1) \leq \overline{c}(P_2)$

The dominance rule limits the number of non-dominated label by $2^{\Delta(NG)-1}$ in each bucket. Considering the whole matrix, there are at most $2^{\Delta(NG)-1}|N|Q$ non-dominated labels in it. This may be reasonable fast depending how large is $\Delta(NG)$.

The complete algorithm for the exact version of the pricing problem with *ng*-route relaxation is shown in Algorithm III.2.

Algorithm III.2 Exact Pricing

procedure EXACTPRICING($\mathcal{M}, NG, \overline{c}$)
  Input: The matrix $\mathcal{M}$, ng-sets $NG_i \subseteq N, \forall i \in V$ and the array of reduced costs $\overline{c}$
  for $q = 1, \ldots, Q$ **do**
    for $i \in V$ **do**
      if $q - d_i > 0$ **then**
        for $j \in N$ **do**
          for $\mathcal{L}(P') \in \mathcal{M}(q - d_i, j)$ **do**
            if $i \notin \Pi(P')$ **then**
              $\mathcal{L}(P) \leftarrow (\overline{c}(P') + \overline{c}_{ij}, i, q, \Pi(P') \cap NG_i \cup \{i\})$
              insertLabel $\leftarrow$ **true**
              for $\mathcal{L}(P'') \in \mathcal{M}(q, i)$ **do**
                if $\mathcal{L}(P)$ *dominates* $\mathcal{L}(P'')$ **then**
                  delete $\mathcal{L}(P'')$
                else if $\mathcal{L}(P'')$ *dominates* $\mathcal{L}(P)$ **then**
                  insertLabel $\leftarrow$ **false**
                  break
              if *insertLabel* **then**
                $\mathcal{M}(q, i) \leftarrow \mathcal{M}(q, i) \cup \mathcal{L}(P)$

Next, we present three techniques to speed up the exact pricing. The first one is the Decremental State Space Relaxation (DSSR), introduced by Righini and Salani in [23] for the pricing with elementary routes and further adapted to the *ng*-route relaxation by Martinelli et al. [15]. The second one is called the completion bounds and finally the last one is a heuristic pricing.

(a) Decremental State-Space Relaxation

The Decremental State-Space Relaxation consists in relaxing totally or partially the *ng*-route restrictions, allowing the occurrence of some cycles which would not appear in a regular run of the algorithm (depending on the level of relaxation, it may allow any cycle to appear). The algorithm then iteratively removes some relaxations until a feasible *ng*-route is found. This way, each iteration of the DSSR pricing sub-problem provides a lower bound on the solution of the original pricing with the complete *ng*-sets.

The relaxation is done by using sets $\Gamma_v^k \subseteq NG_v$, where k is the index of current iteration of the DSSR algorithm. At first, each Γ_v^0 may be empty (meaning that any cycle may happen in the solution) and at each iteration, it checks the best route found by the pricing algorithm w.r.t. the regular *ng*-sets NG_v. If the best route found is *ng*-feasible (or if there is no route with negative reduced cost), the algorithm stops and return this route (resp. none) and any other *ng*-feasible route with negative reduced cost. On the other hand, if the best route violates any *ng*-set, it reduces the relaxation by updating $\Gamma_u^{k+1} = \Gamma_u^k \cup \{w\}$, for all customers $u \in N$ present in the forbidden cycle, where $w \in N$ is the customer on which the cycle happens.

The Algorithm III.3 presents a pseudo code to this framework. The *exactPricing* is the same algorithm showed in III.2 and *selectRoutes* selects routes with negative reduced costs.

Algorithm III.3 Decremental State-Space Relaxation

```
1:  procedure DSSR(M, NG)
2:      Input: The matrix M, ng-sets NG_i ⊆ N, ∀i ∈ N
3:      Γ_i ← ∅, ∀i ∈ N, ng ← false, k ← 0
4:      while not ng do
5:          R ← exactPricing(M, Γ)
6:          R*_k ← selectRoutes(R)
7:          if isNGRoute(R*_k, NG) then
8:              ng ← true
9:          else
10:             updateNGSet(R*_k, NG, Γ)
11:         k ← k + 1
12:     return R*_k
```

(b) Completion bounds

The Completion Bounds is another technique devised to speed up the pricing sub-problem running time. It works within the DSSR and it is similar to the dominance rule in the sense that it also eliminates labels. Given the fact that at an iteration of the DSSR algorithm any previous iteration is a relaxation of the

current one, the dynamic programming matrix of a previous iteration provides valid lower bounds on the reduced cost values for all the paths. This information may be used to avoid an extension of a path by checking if its current reduces cost value plus the lower bound for the remaining path would lead to a non-negative value. Thus, at the end of each DSSR iteration k, the best lower bound for each customer $v \in N$ with every capacity $q \leq Q$ is calculated, in order to be used during iteration $k+1$. These lower bounds are then stored in an auxiliary matrix $\widehat{T}$, which is filled with the value of the best path that starts at a customer i and ends at the depot with total capacity at most q. This process is shown in Equation III.6

$$\widehat{T}_k(q,i) = \min_{q' \leq q}(T_k^*(q,i)) \tag{III.6}$$

After the matrix $\widehat{T}$ is filled, its values may be used in the next iteration to avoid the extension to a new labels $\mathcal{L}(P) = (v(P), q(P), \Pi(P), \bar{c}(P))$. A label is discarded if the following condition hold:

$$\bar{c}(P) + \bar{c}_{ij} + \widehat{T}(Q - q(P), v(P)) \geq 0 \tag{III.7}$$

It is clear that when the left-hand side of Equation III.7 is greater or equal than zero, the label $\mathcal{L}(P)$ cannot generate any routes with negative reduced cost, and then it may be eliminated. This will make the exact pricing converges faster, especially because the lower bounds found become stronger on each new iteration of the DSSR algorithm.

(c) Heuristic Pricing

As a further improvement to speed up the whole column generation procedure, any heuristic pricing approach may be used before running the exact dynamic programming algorithm. The idea is to find as many routes with negative reduced cost as possible using a faster algorithm, thus reducing the number of calls to the exact algorithm. In this dissertation, we use a simplification of the exact algorithm described in Section III.1. Instead of maintaining a list of paths in each dynamic programming matrix $\mathcal{M}(q,v)$ entry, we keep only the path with best reduced cost so far, discarding any other labels which may appear. This modification may seem quite simple at first, but one may notice that when used, this approach may find up to 90% of the total columns generated during the column generation, obtaining a drastic improvement on the overall running time of the solution procedure.

To exemplify how the algorithm works, when we are filling the bucket $\mathcal{M}(q,i)$, if $q - d_i > 0$ we have to look at all buckets in row $\mathcal{M}(q - d_i, i)$, looking for those which may improve the reduced cost and perform a proper extension, with respect to ng-routes. In other words, we want the best bucket where $i \notin \Pi(P)$. The Algorithm III.4 shows a pseudo-code for this procedure.

Analogously to the exact pricing algorithm, after running the pricing, a procedure must be executed to build the best ng-routes found. This procedure will

Algorithm III.4 Heuristic Pricing

procedure HEURISTICPRICING($\mathcal{M}, NG, \overline{c}$)
  Input: The matrix $\mathcal{M}$, ng-sets $NG_i \subseteq V, \forall i \in V$ and the matrix of reduced costs $\overline{c}$
  for $q = 1, \ldots, Q$ **do**
    for $i \in V$ **do**
      $\mathcal{L}(P) \leftarrow (\infty, i, q, -)$
      if $q - d_i > 0$ **then**
        for $j \in N$ **do**
          $\mathcal{L}(P') \leftarrow \mathcal{M}(q - d_i, j)$
          if $i \notin \Pi(P')$ *and* $\overline{c}(P') + \overline{c}_{ij} < \overline{c}(P)$ **then**
            $\mathcal{L}(P) \leftarrow (\overline{c}(P') + \overline{c}_{ij}, i, q, \Pi(P') \cap NG_i \cup \{i\})$
        $\mathcal{M}(q, i) \leftarrow \mathcal{L}(P)$

iterate over all buckets of the matrix $\mathcal{M}$, sorting them by its reduced cost and checking if the reduced cost remains negative when taking the path back to the depot.

Finally, in order to prove the optimality on the last iteration, a run of the complete exact pricing still must be performed. This happens because it not all paths are taken into consideration by the heuristic and there is a chance that feasible routes with negative reduced cost are not built. Moreover, it is noteworthy to mention that the complexity of the heuristic pricing compares to the one of the q-route relaxation, i.e., it is weakly $\mathcal{NP}$-hard, being performed in the pseudo-polynomial time of $\mathcal{O}(Q|N|^2)$.

IV
Cuts

In this chapter we present some cuts to the CVRP. Cuts are used to improve the lower bounds quality. They can be described as valid inequalities which are added to the mathematical formulation when violated by the current solution. The procedure of identification of such violated cuts is called the cut separation algorithm. When dealing with column generation, we classify the cuts in two types: robust cuts and strong cuts. Regardless the name, they differ basically in how the dual variables associated to them can be translated into reduced cost to the pricing. Robust cuts have the particularity that the contribution of their dual variables to the computation of the reduced costs of paths can be decomposed along the edges, and therefore they may be considered in the pricing algorithm without compromising its performance, regardless of the number of robust cuts added. On the other hand, strong cuts requires additional resources in the labels, typically one resource per active cut, which is represented by one additional dimension. Those dimensions are used to control whether an extension should be reward/penalized with the value of the dual variable associated to the cut. Therefore, it is straightforward to notice that with the increase on the number of strong cuts, the pricing loses performance.

During the last 15 years, many different families of both robust and strong cuts were presented for the CVRP. In this work, we consider one family of robust cuts, the Capacity Cuts [24], and one family of strong cuts, the Subset-Row Cuts [20].

IV.1 Capacity Cuts

The Capacity Cuts are a family of robust cuts originated by the generalization of the famous Sub-tour Elimination Cuts for the TSP. They are similar to the capacity constraint showed in Equation (II.4). It is know that to calculate $r(s)$ exactly one needs to solve the Bin Packing Problem which is a strongly $\mathcal{NP}$-hard problem [22]. However, the inequality remains valid if one replaces $r(S)$ on the right-hand side with the lower bound $k(S) = \lceil q(S)/Q \rceil$. Using the relation shown in Section II.2, when we presented the decomposition of the Two Index Formulation into the Set Partitioning Formulation, we can state the Rounded Capacity Cuts as in Equation (IV.1).

$$\sum_{r\in\Omega}\sum_{e\in\delta(S)} b_r^e \lambda_r \geq 2k(S), S \subseteq N \tag{IV.1}$$

To separate these cuts, we use the CVRPSEP package, presented in [25]. Moreover, as mentioned earlier, being robust these cuts do not change the pricing sub-problem complexity. This is due to the possibility of mapping the dual variables associate to them directly into the edges of the original graph. Thus, using this mapping, the dual variables may be directly introduced in Equation (III.2).

IV.2 Subset Row Cuts

The Subset-Row Cuts (SRCs) are a family of strong cuts introduced in 2008 by [20] and since then some works have shown they lead to good improvements on lower bounds values [9, 18]. Their general form is a Chvátal-Gomory Rank-1 Cut, obtained from the Constraints (II.19) of the SP formulation. Given a subset of customers $C \subseteq N$ and a multiplier $p \in \mathbb{R}, 0 < p < 1$, they are shown in Equation (IV.2).

$$\sum_{r\in\Omega}\left\lfloor p\sum_{i\in C} a_i^r \right\rfloor \lambda_r \leq \lfloor p\,|C| \rfloor \tag{IV.2}$$

Some results for different choices of $(|C|, p)$-SRCs are published in the literature and the $(3, 1/2)$-SRCs are shown to be the most promising ones. Their definition is presented in Equation (IV.3).

$$\begin{aligned}\sum_{r\in\Omega}\left\lfloor \frac{1}{2}\sum_{i\in C} a_i^r \right\rfloor \lambda_r &\leq \left\lfloor \frac{3}{2} \right\rfloor \\ \sum_{r\in\Omega}\left\lfloor \frac{1}{2}\sum_{i\in C} a_i^r \right\rfloor \lambda_r &\leq 1\end{aligned} \tag{IV.3}$$

Despite their success, the SRCs are still hard to separate (the only known separation is to enumerate all sets C) and they are also hard to be considered in the pricing sub-problem. Each new SRC added to the SP formulation introduces a new resource for the pricing sub-problem. In the case of $(3, 1/2)$-SRCs, for each cut $c \in \mathcal{C}$ (where $\mathcal{C}$ is the set of all separated SRCs) there must exist an integer counter sr_c on each label which is incremented every time the path reaches a customer $v \in C$. When this counter reaches 2, the dual variable σ_c associated to the cut must be subtracted to the path's current reduced cost value and the counter returns to 0.

In addition to the above modification, the dominance rule described in Section III.2 must also be changed to consider the reduced cost adjustment. When testing if a given label $\mathcal{L}(P_1)$ dominates $\mathcal{L}(P_2)$, the impact of future contributions of dual variables σ_c must be taken into account. Since $\sigma_c \leq 0$, its introduction represents a penalization for an extension and this weaken the dominance rule. This means that when $sr_c(P_1) > sr_c(P_2)$, in the worst-case the contribution of σ_c for this SRC may

occur only for path P_1 and the dominance rule must consider this case. Therefore, the new dominance rule turns into the following:

(i) $v(P_1) = v(P_2)$

(ii) $q(P_1) \leq q(P_2)$

(iii) $\Pi(P_1) \subseteq \Pi(P_2)$

(iv) $\bar{c}(P_1) \leq \bar{c}(P_2) + \sum_{c \in \mathcal{C}: sr_c(P_1) > sr_c(P_2)} \sigma_c$

IV.3 Limited Memory Subset Row Cut

Recently, a relaxation to the SRCs was introduced by [19] called the Limited Memory Subset-Row Cuts (lm-SRCs). The intuition is analogous to the *ng*-route relaxation. Each customer now has a memory set M_v containing the SRCs it "remembers". Thus, every time a path P with $sr_c(P) > 0$ reaches a customer with $c \notin M_v$, the path "forgets" the lm-SRC c by setting $sr_c(P) = 0$. If we were to consider an SRC in its original form, it would be the same as adding this cut to the memory set of all customers.

When creating a new lm-SRC, it may be added to the memory set of any customer. Nevertheless, there is a subset of customers required to remember the new lm-SRC for it to be violated by the current solution. These are the ones appearing from an odd visit to set C until the following visit, on every route considered during the separation routine and besides the ones already in set C. For example, suppose $C = \{1, 2, 3\}$, $r_1 = (0, 1, 4, 5, 3, 7, 4, 0)$, $r_2 = (0, 8, 2, 8, 6, 2, 8, 0)$. Let $\lambda_{r_1} > 0$ and $\lambda_{r_2} > 0$ in a way this solution violates a $(3, 1/2)$-SRC. Considering route r_1, at the moment the customer 1 is visited, the counter of the cut will be incremented. When visiting customer 3, the counter will be incremented again and the cut will be added to the memory set of all vertices between 1 and 3. Repeating this process for the second route, the cut will be added to the memory set M_v of customers $v \in \{1, 2, 3\} \cup \{4, 5\} \cup \{8, 6\}$.

The consequence of the relaxation is that now some routes which were considered in the original SRC are not considered in the relaxed lm-SRC anymore. If we define $\alpha_r(C, p, M)$ as being the coefficient of a route r on the new lm-SRC, Equation (IV.4) shows its new definition.

$$\sum_{r \in \Omega} \alpha_r(C, p, M)\lambda_r \leq \lfloor p\,|C| \rfloor \tag{IV.4}$$

When $M = N$ the lm-SRC will be exactly as in the SRC and the function α_r will return $\lfloor p \sum_{i \in C} a_i^r \rfloor$. On the other hand, if $|M| < |N|$ the function α_r will return a smaller coefficient for some routes in the inequality. This happens because every time a route r leaves M, the variable state is reset to zero. So, only vertices in the memory set will keep the state of the cut. For those which are not in M, the state

of the cut is zero by definition. It means that the duals variables σ_s associated to the cuts will only play a role along extensions among vertices in M.

The main difference between the SRCs and the lm-SRCs is that the new version only tracks the state of the cuts among a subset of the customers sets and because of that fewer extensions will be penalized. This makes fewer buckets to be affected and the dominance will be improved. This can be noticed in practice by a speed up on the labeling algorithm by a factor $\Theta(n/M_{avg})$, where M_{avg} is the average size of the memory sets. Consequently, more lm-SRCs may be added to the formulation before the pricing loses performance.

(a) Separation

The first step of the lm-SRCs separation is to identify every subset of customers C which violate the Equation IV.4. This step can be done enumerating all subsets of customers of size three. To do that, we consider all λ_r variables with positive value from the solution of the restricted master problem and the routes associated with those variables. Then, we check for each route which triplets had at least two customer visited. Then, we verify if the left hand side of the Equation (IV.4) would be larger than one, which consists of a violation. Algorithm IV.1 illustrates how the separation is done. Let a_i^r keep the same notation as seen before – it represents the number of times a customer i was visited by route r.

Algorithm IV.1 Algorithm to separate lm-srcs

```
1:  procedure SEPARATION(λ_r, routesInSolution)
2:      violations ← {}
3:      for i ∈ N do
4:          for j ∈ N \ {i} do
5:              for k ∈ N \ {i, j} do
6:                  violation ← new Violation
7:                  for r ∈ routesInSolution do
8:                      if Σ_{c∈C} a_c^r > 1 then
9:                          violation.triplet ← {i, j, k}
10:                         violation.routes ← violation.routes ∪ {r}
11:                         violation.lhs ← violation.lhs + ⌊½ Σ_{c∈C} a_c^r λ_r⌋
12:                 if violation.lhs > 1 then
13:                     violations ← violations ∪ violation
14:     return violations
```

Algorithm IV.1 tests if a violation occurs for each possible triplet. Note that are, in this case, $\binom{N}{3}$ triplets. So it is necessary to keep in mind to not return all cuts found to the master problem in order to minimize the impact on the pricing. In this work, we setup the limit of 500 cuts at each time the separation is called.

Knowing which triplets were violated, we now calculate the cut memory M. This minimal set represents which customers will remember a certain cut during

the pricing execution. Initially, $M = C$ and the memory set is updated with every customer between two visits to a vertices in C, as described earlier. This process is describe in Algorithm IV.2, where we associate a state variable to control the parity of visits to vertices in the triplet.

Algorithm IV.2 Algorithm to calculate the cut memory

procedure CALCULATEM(C, λ, p)
  $M \leftarrow C$
  for each route r where $\lambda_r > 0$ and $\lfloor p \sum_{i \in C} a_i^r \rfloor > 0$ **do**
    $state \leftarrow 0$, $Aux \leftarrow \emptyset$
    for each vertex i in route r (in order) **do**
      if $i \in C$ **then**
        $state \leftarrow state + p$
        if $state \geq 1$ **then**
          $M \leftarrow M \cup Aux, Aux \leftarrow \emptyset, state \leftarrow state - 1$
      else if $state > 0$ **then**
        $Aux \leftarrow Aux \cup \{i\}$
  return M

Finally, we just have to calculate the route coefficients to properly add the cut into the formulation. This process is done to every route in set Ω, regardless its solution value, even if it is zero. As mentioned previously, the route coefficient in the lm-SRC will be smaller or equal to one in SRC. Algorithm IV.3 shows how to calculate these coefficients.

Algorithm IV.3 Algorithm to calculate the coefficient of a route within a cut

procedure CALCULATEALPHA(C, M, p, r)
  $coef \leftarrow 0, state \leftarrow 0$
  for each vertex v in route r (in order) **do**
    if $i \notin M$ **then**
      $state \leftarrow 0$
    else if $i \in C$ **then**
      $state \leftarrow state + p$
      if $state \geq 1$ **then**
        $coef \leftarrow coef + 1, state \leftarrow state - 1$
  return $coef$

(b) Pricing changes

Here is where the lm-SRC shows its effectiveness when comparing with regular SRC. Both families of cuts add a new resource to the label to track the state of each active cut. Therefore, the labels becomes $\mathcal{L}(P) = (\overline{c}(P), v(P), q(P), \Pi(P), sr_c(P))$ where $sr_c(P)$ is the vector of states corresponding to n_S lm-SRCs with non-zero duals variables in the current master LP solution.

Since $\sigma_c < 0$, the consideration of a SRC represents a penalization for one extension and this weakens the dominance rule. In fact, the dominance rule presented in Section IV.2 remains the same, but now, as the pricing "forgets" some lm-SRCs, condition (iv) of the dominance rule will be strengthened.

Another change in the pricing is when performing an extension $j \to i$, we have to check each lm-SRC c that j remembers, if $i \in M(c)$. If that is not the case, c will be forgotten, which means that the state variable of the cut will be set as zero. On the other hand if $i \in M(c)$, $i \in C(c)$ and $state \geq 1$ then the dual variable σ_c should penalize this extension. To illustrate the idea, we present Algorithm IV.4

Algorithm IV.4 Exact Pricing

procedure EXACTPRICINGSRC($\mathcal{M}, NG, \overline{c}, s$)
  Input: The matrix $\mathcal{M}$, ng-sets $NG_i \subseteq N, \forall i \in V$, the array of reduced costs $\overline{c}$ and s vector with actives lm-SRC
  for $q = 1, \ldots, Q$ **do**
    for $i \in V$ **do**
      if $q - d_i > 0$ **then**
        for $j \in N$ **do**
          for $L(P') \in \mathcal{M}(q - d_i, j)$ **do**
            if $i \notin \Pi(P')$ **then**
              $\overline{c}(P) \leftarrow \overline{c}(P') + \overline{c}_{ij}$
              $S(P) = S(P')$
              for $c = 1$ **to** n_s **do**
                if $i \notin M(c)$ **then**
                  $sr_c(P) \leftarrow 0$
                else if $i \in C(c)$ **then**
                  $sr_c(P) \leftarrow sr_c(P) + 1$
                  if $sr_c(P) \geq 1$ **then**
                    $\overline{c}(P) \leftarrow \overline{c}(P) - \sigma_c$
                    $sr_c(P) \leftarrow sr_c(P) - 1$
              $\mathcal{L}(P) \leftarrow (\overline{c}(P') + \overline{c}_{ij}, i, q, \Pi(P') \cap N_i \cup \{i\}, sr_c(P))$
              insertLabel $\leftarrow$ **true**
              for $\mathcal{L}(P'') \in M(q, i)$ **do**
                if $\mathcal{L}(P)$ *dominates* $\mathcal{L}(P'')$ **then**
                  delete $\mathcal{L}(P'')$
                else if $\mathcal{L}(P'')$ *dominates* $\mathcal{L}(P)$ **then**
                  insertLabel $\leftarrow$ **false**
                  break
              if *insertLabel* **then**
                $\mathcal{M}(q, i) \leftarrow M(q, i) \cup \mathcal{L}(P)$

V
Computational Experiments

In this chapter we report a detailed computational experiment for a large set of CVRP instances, comparing the leverage obtained with the lm-SRC combined with the techniques presented in this thesis. Furthermore, we present a comparison between the bounds found by our approaches and those found by Pecin et al. [19]. For our computational experiments, we use the standard classes of instances (A, B, E, M, and P) available at CVRPLib (http://vrp.galgos.inf.puc-rio.br).

The algorithms were implemented in C++, using Microsoft Visual Studio 2013 and Gurobi 6.5 [26]. The experiments were conducted on a Intel Core i7-3960X 3.30GHz with 64GB RAM running Linux Ubuntu Server 14.04.

We start presenting the comparison between a simple version using only the Set Partitioning Formulation, *ng*-routes and the additional lm-src. We setup the size of the *ng*-sets to be 8 for all tests. The results are presented in Table V.1. Columns **Routes** represent the number of routes that were generated during the process, columns **Heu** and **Exa** show the amount of time spent by the heuristic pricing and the exact pricing, respectively. Column **Sep** shows the total number of separated lm-SRCs and column **Act** shows the number of lm-SRCs which were active in the last resolution of the formulation. In order to keep the formulation more compact, after running the heuristic pricing, if any lm-SRC was not active, we remove it from the formulation. Lines with dash instead of values reached the time limit that we setup as 2 hours. At the end of each table, we calculate the average of each column for those instances that had finished in both configuration, with and without the consideration of the lm-SRC.

Other strategies were used in the attempt to improve the running time. We limited the number of routes that can be returned by the pricing in 20. As said before, we also limited the number of strong cuts that are added into the formulation at each iteration to be at most 500. Finally, we tried to minimize the number of runs of the exact pricing. To do that, first we run the heuristic pricing until no routes are found, then we separate the cuts and if no cuts were found we run the exact pricing.

Looking closer to Table V.1, we can immediately see the power of the lm-SRC. The number of instances which was equal to the known lower bound increased from 2 to 25, however the number of instances which do not finished within the two hour time limit was 21 out of 93 instances tested and the elapsed time of the remaining instances increased significantly.

In Table V.2 we show the results of the comparison of a version using *ng*-routes with DSSR to the one with the addition of the lm-SRCs. The number of instances which reached the known lower bound was the same, however the number of instances that were timed out decreased from 25 to 21. In this test we used the DSSR technique as presented in Section III.2(a).

The following test combined *ng*-routes, DSSR and Completion Bounds and compares the results with and without lm-SRCs. They are presented in Table V.3. In this test, we did not notice any relevant changes from the one only considering DSSR.

In Table V.4 we combine *ng*-routes with the Capacity Cuts, presented in Section IV.1. Capacity cuts had a great role, increasing the number of instances that reached the known lower bound to 18. Combining them with the lm-SRCs also showed a great improvement and the instances that reached the known lower bound were increased to 53 and the number of instances timed out decrease to 7.

Finally, we tested the combination of all techniques presented in this thesis with the addition of lm-SRCs. In Table V.5 we show the results that we found. In this test, we saw the number of instances that reached the knwon lower bound to remain the same as those tested with only the Capacity Cuts, with and without the lm-SRCs. We also show the comparison with the lower bound found by Pecin et al. [19]. These results are presented in Table V.6.

Table V.1: Comparison over a pure version

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
A-n32-k5	784	770.286	0.872	999	60	3	784	17.764	1713	117	7	458	33
A-n33-k5	661	653.727	0.42	771	46	2	661	1.584	933	64	3	478	29
A-n33-k6	742	732.1	0.383	791	46	2	738.333	2.012	1047	100	4	863	27
A-n34-k5	778	746.012	0.475	885	48	1	767.117	12.109	1756	173	6	1392	83
A-n36-k5	799	776.276	0.797	1230	71	2	796.5	1999.12	2537	200	11	2644	84
A-n37-k5	669	657.811	1.139	1504	82	3	669	15.381	2340	159	4	1224	30
A-n37-k6	949	925.407	0.473	900	50	2	939.488	8.057	1462	169	6	1528	60
A-n38-k5	730	695.417	0.696	1027	57	3	726.273	14.34	1798	139	5	2050	59
A-n39-k5	822	799.842	1.066	1328	78	4	817.102	82.229	2199	198	8	2219	117
A-n39-k6	831	806.672	0.866	1209	68	3	827.473	216.208	2402	292	10	2694	128
A-n44-k6	937	926.641	0.711	1082	59	2	933.914	8.797	1516	165	5	1978	79
A-n45-k6	944	927.25	0.941	1276	71	2	944	7.607	1743	114	5	1000	46
A-n45-k7	1146	1124.67	0.739	1228	66	1	1133.05	4.277	1556	141	3	1452	58
A-n46-k7	914	904.626	1.107	1484	79	2	914	4.777	1814	106	5	495	39
A-n48-k7	1073	1053.08	1.209	1308	73	3	1073	7.925	1669	111	6	1000	48
A-n53-k7	1010	992.378	2.395	2151	119	3	—	—	—	—	—	—	—
A-n54-k7	1167	1137.06	1.489	1522	82	3	1159.6	59.922	2695	253	9	3199	127
A-n55-k9	1073	1059.03	1.151	1366	72	3	1070.67	11.189	2069	178	6	3816	81
A-n60-k9	1354	1323.32	1.726	1816	94	2	1339.03	31.824	2824	243	7	4288	139
A-n61-k9	1034	1010.24	1.418	1537	84	2	1031.33	33.25	2439	244	8	3273	136
A-n62-k8	1288	1250.24	3.093	2184	118	5	1274.37	53.241	3322	307	9	6054	129
A-n63-k10	1314	1286.58	1.944	1798	94	3	1299.6	18.276	2676	201	6	3067	84

Continued on next page

Table V.1: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
A-n63-k9	1616	1579.13	1.463	1547	81	2	1602.61	46.092	2731	265	7	3903	140
A-n64-k9	1401	1368.24	2.212	1924	105	3	1391.02	2753.64	3247	275	9	4228	128
A-n65-k9	1174	1147.33	1.76	1784	94	2	1169.43	47.386	3163	307	7	4399	146
A-n69-k9	1159	1129.97	2.85	2190	118	3	1150.53	68.922	3324	258	8	3129	147
A-n80-k10	1763	1729.81	4.969	2977	155	3	1748.09	715.984	4633	412	9	5872	176
B-n45-k5	751	688.901	2.407	1846	103	4	—	—	—	—	—	—	—
B-n41-k6	829	797.722	0.638	889	50	2	814.604	13.529	1439	160	7	2574	102
B-n39-k5	549	521.071	1.988	1407	86	6	—	—	—	—	—	—	—
B-n38-k6	805	741.634	0.947	1075	64	2	—	—	—	—	—	—	—
B-n35-k5	955	868.083	0.963	1176	70	3	926.018	345.894	3414	382	14	2353	136
B-n34-k5	788	755.231	0.996	1209	72	5	773.636	50.469	2179	220	11	2991	98
B-n31-k5	672	661.207	0.411	774	44	2	666.191	5.649	1207	121	6	1309	56
B-n57-k7	1153	1126.77	2.604	1881	99	4	—	—	—	—	—	—	—
B-n56-k7	707	635.601	3.727	1830	111	7	—	—	—	—	—	—	—
B-n52-k7	747	687.348	2.168	1601	95	5	—	—	—	—	—	—	—
B-n51-k7	1032	964.287	1.682	1351	74	4	—	—	—	—	—	—	—
B-n50-k8	1312	1266.45	1.905	1188	70	4	1282.79	38.627	1753	207	9	3178	111
B-n50-k7	741	690.528	1.83	1570	89	2	—	—	—	—	—	—	—
B-n45-k6	678	652.65	1.413	1043	65	4	666.834	50.152	1912	254	7	4386	130
B-n44-k7	909	865.181	0.738	991	55	2	880.067	568.848	2098	255	10	2892	131
B-n43-k6	742	703.674	1.048	1142	63	3	723.319	515.204	3149	353	15	3748	152
B-n78-k10	1221	1166.73	4.913	2689	141	3	—	—	—	—	—	—	—
B-n68-k9	1272	1198.2	3.135	2249	120	2	—	—	—	—	—	—	—

Continued on next page

Table V.1: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
B-n67-k10	1032	999.204	2.455	2113	111	2	1028.42	146.339	3684	310	11	4632	132
B-n66-k9	1316	1257.22	2.862	2077	111	4	—	—	—	—	—	—	—
B-n64-k9	861	808.253	2.777	1959	105	3	—	—	—	—	—	—	—
B-n57-k9	1598	1571.99	1.681	1550	85	3	1592.7	52.658	2707	219	7	2768	123
B-n63-k10	1496	1456.86	1.608	1634	88	2	1496	330.42	3381	294	11	3615	121
E-n76-k8	735	717.819	4.774	2680	140	2	732.929	297.045	4775	408	9	4048	195
E-n76-k7	682	664.202	8.047	3176	171	3	—	—	—	—	—	—	—
E-n76-k14	1021	1001.85	1.18	1288	67	2	1014.75	7.432	1627	145	4	1363	106
E-n76-k10	830	811.767	2.258	1903	98	1	825.787	28.103	2774	211	4	2535	160
E-n51-k5	521	517.135	1.991	1703	92	2	521	9.098	2008	128	5	500	32
E-n33-k4	835	820.918	3.153	1351	76	3	831.484	74.362	2611	212	5	1597	84
E-n30-k3	534	478.709	1.448	1339	89	3	—	—	—	—	—	—	—
E-n23-k3	569	564.413	5.989	944	68	2	569	12.535	977	87	2	706	8
E-n22-k4	375	373.875	0.095	357	23	1	375	0.112	361	25	1	86	3
E-n101-k8	815	789.372	16.688	4694	243	3	—	—	—	—	—	—	—
E-n101-k14	1067	1047.2	5.102	2889	147	2	1061.12	62.95	4028	291	6	3909	211
E-n31-k7	379	378	0.486	1015	59	1	379	0.613	1037	65	1	126	8
E-n13-k4	247	247	0.014	73	10	1	247	0.014	73	10	1	0	0
P-n55-k10	694	680.06	0.656	1023	55	1	689.193	5.335	1367	147	3	1224	100
P-n23-k8	529	529	0.016	563	4	1	529	0.016	563	4	1	0	0
P-n50-k10	696	687.347	0.461	812	45	2	694.223	1.435	969	84	2	433	53
P-n45-k5	510	502.349	1.253	1506	82	1	510	12.715	2234	145	4	1175	50
P-n40-k5	458	452.5	0.864	1294	68	1	458	4.648	1633	105	4	583	28

Continued on next page

Table V.1: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
P-n22-k8	603	601.25	0.021	146	11	1	603	0.036	155	16	2	16	3
P-n22-k2	216	214.5	0.592	883	64	2	216	1.356	1015	83	3	130	5
P-n21-k2	211	210.556	0.366	670	52	1	211	0.522	674	54	1	235	8
P-n20-k2	216	210	0.295	592	47	1	216	1.874	784	77	4	240	13
P-n19-k2	212	209.714	0.275	405	39	3	212	1.416	527	57	5	131	13
P-n16-k8	450	443.667	0.01	59	8	1	450	0.012	61	10	1	5	4
P-n101-k4	681	669.044	111.742	9909	577	9	—	—	—	—	—	—	—
P-n76-k5	627	614.86	16.051	4303	230	5	—	—	—	—	—	—	—
P-n76-k4	593	586.502	27.43	4929	286	7	593	963.096	7437	536	14	3500	73
P-n70-k10	827	809.29	1.522	1581	82	1	823.067	20.438	2358	196	4	2300	155
P-n65-k10	792	783.126	1.294	1334	70	2	792	5.995	1741	112	3	1338	89
P-n60-k15	968	959.606	0.34	784	40	1	967.465	1.098	898	75	2	504	55
P-n60-k10	744	736.528	1.052	1123	61	3	743.094	5.236	1577	119	3	1278	70
P-n55-k7	568	555.362	1.456	1518	80	1	564.192	45.85	2559	198	6	1929	118
P-n55-k15	989	968.361	7.423	162367	6	1	984.457	39.94	162450	38	2	378	53
P-n51-k10	741	732.955	0.383	830	43	1	740.778	1.973	1121	101	3	853	52
P-n50-k8	631	614.641	0.456	796	39	1	625.061	7.783	1249	125	4	1146	126
P-n50-k7	554	545.399	0.881	1222	63	1	554	7.805	1638	110	4	1063	63
M-n151-k12	1015	995.728	46.443	7398	381	4	1009.65	5099.56	11653	927	18	13943	349
M-n121-k7	1034	1026.37	62.187	6878	364	9	—	—	—	—	—	—	—
M-n101-k10	820	818.367	13.344	4644	251	12	820	77.95	6196	427	19	7434	15
M-n200-k16	1274	1250.23	135.503	9450	481	5	—	—	—	—	—	—	—

Continued on next page

Table V.1: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
M-n200-k17	1275	1252.52	135.413	9590	489	5	—	—	—	—	—	—	—
Average	839.271	820.654	2.596	3743.686	78.3	2.357	833.905	216.544	4538.014	189.914	6.157	2226.529	84.957

Table V.2: Comparison using DSSR

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
A-n32-k5	784	770.286	0.818	999	60	3	784	17.836	1713	117	7	458	33
A-n33-k5	661	653.727	0.394	771	46	2	661	1.638	933	64	3	478	29
A-n33-k6	742	732.1	0.378	791	46	2	738.333	2.038	1047	100	4	863	27
A-n34-k5	778	746.012	0.477	885	48	1	767.117	12.45	1756	173	6	1392	83
A-n36-k5	799	776.276	0.842	1230	71	2	796.5	1999.4	2537	200	11	2644	84
A-n37-k5	669	657.811	1.131	1504	82	3	669	15.443	2340	159	4	1224	30
A-n37-k6	949	925.407	0.509	900	50	2	939.488	8.064	1462	169	6	1528	60
A-n38-k5	730	695.417	0.687	1027	57	3	726.273	14.387	1798	139	5	2050	59
A-n39-k5	822	799.842	1.074	1328	78	4	817.102	82.032	2199	198	8	2219	117
A-n39-k6	831	806.672	0.904	1209	68	3	827.473	215.688	2402	292	10	2694	128
A-n44-k6	937	926.641	0.712	1082	59	2	933.914	8.853	1516	165	5	1978	79
A-n45-k6	944	927.25	0.937	1276	71	2	944	7.578	1743	114	5	1000	46
A-n45-k7	1146	1124.67	0.793	1228	66	1	1133.05	4.278	1556	141	3	1452	58
A-n46-k7	914	904.626	1.112	1484	79	2	914	4.848	1814	106	5	495	39
A-n48-k7	1073	1053.08	1.156	1308	73	3	1073	8.018	1669	111	6	1000	48
A-n53-k7	1010	992.378	2.359	2151	119	3	—	—	—	—	—	—	—
A-n54-k7	1167	1137.06	1.481	1522	82	3	1159.6	58.701	2695	253	9	3199	127
A-n55-k9	1073	1059.03	1.157	1366	72	3	1070.67	11.024	2069	178	6	3816	81
A-n60-k9	1354	1323.32	1.768	1816	94	2	1339.03	31.858	2824	243	7	4288	139
A-n61-k9	1034	1010.24	1.432	1537	84	2	1031.33	32.755	2439	244	8	3273	136
A-n62-k8	1288	1250.24	3.108	2184	118	5	1274.37	53.001	3322	307	9	6054	129
A-n63-k10	1314	1286.58	1.929	1798	94	3	1299.6	18.121	2676	201	6	3067	84

Continued on next page

Table V.2: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
A-n63-k9	1616	1579.13	1.466	1547	81	2	1602.61	46.184	2731	265	7	3903	140
A-n64-k9	1401	1368.24	2.215	1924	105	3	1391.02	2754.31	3247	275	9	4228	128
A-n65-k9	1174	1147.33	1.763	1784	94	2	1169.43	48.613	3163	307	7	4399	146
A-n69-k9	1159	1129.97	2.867	2190	118	3	1150.53	69.044	3324	258	8	3129	147
A-n80-k10	1763	1729.81	4.966	2977	155	3	1748.09	716.846	4633	412	9	5872	176
B-n45-k5	751	688.901	2.305	1846	103	4	—	—	—	—	—	—	—
B-n41-k6	829	797.722	0.577	889	50	2	814.604	13.524	1439	160	7	2574	102
B-n39-k5	549	521.071	1.993	1407	86	6	—	—	—	—	—	—	—
B-n38-k6	805	741.634	0.981	1075	64	2	—	—	—	—	—	—	—
B-n35-k5	955	868.083	0.943	1176	70	3	926.018	345.301	3414	382	14	2353	136
B-n34-k5	788	755.231	0.981	1209	72	5	773.636	50.205	2179	220	11	2991	98
B-n31-k5	672	661.207	0.393	774	44	2	666.191	5.652	1207	121	6	1309	56
B-n57-k7	1153	1126.77	2.619	1881	99	4	—	—	—	—	—	—	—
B-n56-k7	707	635.601	3.687	1830	111	7	—	—	—	—	—	—	—
B-n52-k7	747	687.348	2.427	1601	95	5	—	—	—	—	—	—	—
B-n51-k7	1032	964.287	1.581	1351	74	4	—	—	—	—	—	—	—
B-n50-k8	1312	1266.45	1.592	1188	70	4	1282.79	38.502	1753	207	9	3178	111
B-n50-k7	741	690.528	1.977	1570	89	2	—	—	—	—	—	—	—
B-n45-k6	678	652.65	1.584	1043	65	4	666.834	49.616	1912	254	7	4386	130
B-n44-k7	909	865.181	0.783	991	55	2	880.067	567.824	2098	255	10	2892	131
B-n43-k6	742	703.674	1.116	1142	63	3	723.319	514.972	3149	353	15	3748	152
B-n78-k10	1221	1166.73	5.038	2689	141	3	—	—	—	—	—	—	—
B-n68-k9	1272	1198.2	3.129	2249	120	2	—	—	—	—	—	—	—

Continued on next page

Table V.2: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
B-n67-k10	1032	999.204	2.453	2113	111	2	1028.42	147.802	3684	310	11	4632	132
B-n66-k9	1316	1257.22	2.845	2077	111	4	—	—	—	—	—	—	—
B-n64-k9	861	808.253	2.621	1959	105	3	—	—	—	—	—	—	—
B-n57-k9	1598	1571.99	1.676	1550	85	3	1592.7	51.958	2707	219	7	2768	123
B-n63-k10	1496	1456.86	1.581	1634	88	2	1496	330.134	3381	294	11	3615	121
E-n76-k8	735	717.819	4.764	2680	140	2	732.929	299.893	4775	408	9	4048	195
E-n76-k7	682	664.202	8.066	3176	171	3	—	—	—	—	—	—	—
E-n76-k14	1021	1001.85	1.176	1288	67	2	1014.75	7.488	1627	145	4	1363	106
E-n76-k10	830	811.767	2.248	1903	98	1	825.787	27.888	2774	211	4	2535	160
E-n51-k5	521	517.135	1.966	1703	92	2	521	9.046	2008	128	5	500	32
E-n33-k4	835	820.918	3.147	1351	76	3	831.484	75.211	2611	212	5	1597	84
E-n30-k3	534	478.709	1.457	1339	89	3	—	—	—	—	—	—	—
E-n23-k3	569	564.413	5.968	944	68	2	569	12.716	977	87	2	706	8
E-n22-k4	375	373.875	0.093	357	23	1	375	0.112	361	25	1	86	3
E-n101-k8	815	789.372	16.878	4694	243	3	—	—	—	—	—	—	—
E-n101-k14	1067	1047.2	5.086	2889	147	2	1061.12	62.59	4028	291	6	3909	211
E-n31-k7	379	378	0.491	1015	59	1	379	0.616	1037	65	1	126	8
E-n13-k4	247	247	0.014	73	10	1	247	0.014	73	10	1	0	0
P-n55-k10	694	680.06	0.646	1023	55	1	689.193	5.283	1367	147	3	1224	100
P-n23-k8	529	529	0.017	563	4	1	529	0.016	563	4	1	0	0
P-n50-k10	696	687.347	0.46	812	45	2	694.223	1.427	969	84	2	433	53
P-n45-k5	510	502.349	1.256	1506	82	1	510	12.781	2234	145	4	1175	50
P-n40-k5	458	452.5	0.858	1294	68	1	458	4.629	1633	105	4	583	28

Continued on next page

Table V.2: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
P-n22-k8	603	601.25	0.02	146	11	1	603	0.033	155	16	2	16	3
P-n22-k2	216	214.5	0.582	883	64	2	216	1.367	1015	83	3	130	5
P-n21-k2	211	210.556	0.367	670	52	1	211	0.524	674	54	1	235	8
P-n20-k2	216	210	0.297	592	47	1	216	1.829	784	77	4	240	13
P-n19-k2	212	209.714	0.275	405	39	3	212	1.416	527	57	5	131	13
P-n16-k8	450	443.667	0.016	59	8	1	450	0.012	61	10	1	5	4
P-n101-k4	681	669.044	112.183	9909	577	9	—	—	—	—	—	—	—
P-n76-k5	627	614.86	16.071	4303	230	5	—	—	—	—	—	—	—
P-n76-k4	593	586.502	27.423	4929	286	7	593	946.706	7437	536	14	3500	73
P-n70-k10	827	809.29	1.576	1581	82	1	823.067	20.964	2358	196	4	2300	155
P-n65-k10	792	783.126	1.291	1334	70	2	792	6.037	1741	112	3	1338	89
P-n60-k15	968	959.606	0.34	784	40	1	967.465	1.099	898	75	2	504	55
P-n60-k10	744	736.528	1.05	1123	61	3	743.094	5.182	1577	119	3	1278	70
P-n55-k7	568	555.362	1.458	1518	80	1	564.192	45.917	2559	198	6	1929	118
P-n55-k15	989	968.361	7.578	162367	6	1	984.457	39.951	162450	38	2	378	53
P-n51-k10	741	732.955	0.385	830	43	1	740.778	2.039	1121	101	3	853	52
P-n50-k8	631	614.641	0.458	796	39	1	625.061	7.728	1249	125	4	1146	126
P-n50-k7	554	545.399	0.882	1222	63	1	554	7.806	1638	110	4	1063	63
M-n151-k12	1015	995.728	46.326	7398	381	4	1009.65	5098.44	11653	927	18	13943	349
M-n121-k7	1034	1026.37	62.104	6878	364	9	—	—	—	—	—	—	—
M-n101-k10	820	818.367	12.802	4644	251	12	820	75.618	6196	427	19	7434	15
M-n200-k16	1274	1250.23	134.12	9450	481	5	—	—	—	—	—	—	—

Continued on next page

Table V.2: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
M-n200-k17	1275	1252.52	136.099	9590	489	5	—	—	—	—	—	—	—
Average	839.271	820.654	2.587	3743.686	78.3	2.357	833.905	216.298	4538.014	189.914	6.157	2226.529	84.957

Table V.3: DSSR + CB

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
A-n32-k5	784	770.286	0.848	999	60	3	784	16.215	1713	117	7	458	33
A-n33-k5	661	653.727	0.418	771	46	2	661	1.684	933	64	3	478	29
A-n33-k6	742	732.1	0.37	791	46	2	738.333	1.922	1047	100	4	863	27
A-n34-k5	778	746.012	0.484	885	48	1	767.117	12.114	1756	173	6	1392	83
A-n36-k5	799	776.276	0.844	1230	71	2	796.5	1999.7	2537	200	11	2644	84
A-n37-k5	669	657.811	1.143	1504	82	3	669	15.357	2340	159	4	1224	30
A-n37-k6	949	925.407	0.473	900	50	2	939.488	8.104	1462	169	6	1528	60
A-n38-k5	730	695.417	0.687	1027	57	3	726.273	14.442	1798	139	5	2050	59
A-n39-k5	822	799.842	1.071	1328	78	4	817.102	81.826	2199	198	8	2219	117
A-n39-k6	831	806.672	0.864	1209	68	3	827.473	216.221	2402	292	10	2694	128
A-n44-k6	937	926.641	0.712	1082	59	2	933.914	8.862	1516	165	5	1978	79
A-n45-k6	944	927.25	0.992	1276	71	2	944	7.597	1743	114	5	1000	46
A-n45-k7	1146	1124.67	0.738	1228	66	1	1133.05	4.276	1556	141	3	1452	58
A-n46-k7	914	904.626	1.105	1484	79	2	914	4.797	1814	106	5	495	39
A-n48-k7	1073	1053.08	1.145	1308	73	3	1073	7.932	1669	111	6	1000	48
A-n53-k7	1010	992.378	2.365	2151	119	3	—	—	—	—	—	—	—
A-n54-k7	1167	1137.06	1.518	1522	82	3	1159.6	59.182	2695	253	9	3199	127
A-n55-k9	1073	1059.03	1.202	1366	72	3	1070.67	11.157	2069	178	6	3816	81
A-n60-k9	1354	1323.32	1.738	1816	94	2	1339.03	31.832	2824	243	7	4288	139
A-n61-k9	1034	1010.24	1.432	1537	84	2	1031.33	33.241	2439	244	8	3273	136
A-n62-k8	1288	1250.24	3.113	2184	118	5	1274.37	53.084	3322	307	9	6054	129
A-n63-k10	1314	1286.58	2.257	1798	94	3	1299.6	18.253	2676	201	6	3067	84

Continued on next page

Table V.3: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
A-n63-k9	1616	1579.13	1.473	1547	81	2	1602.61	46.1	2731	265	7	3903	140
A-n64-k9	1401	1368.24	2.216	1924	105	3	1391.02	2753.11	3247	275	9	4228	128
A-n65-k9	1174	1147.33	1.757	1784	94	2	1169.43	48.122	3163	307	7	4399	146
A-n69-k9	1159	1129.97	2.882	2190	118	3	1150.53	70.723	3324	258	8	3129	147
A-n80-k10	1763	1729.81	4.944	2977	155	3	1748.09	714.275	4633	412	9	5872	176
B-n45-k5	751	688.901	2.394	1846	103	4	—	—	—	—	—	—	—
B-n41-k6	829	797.722	0.585	889	50	2	814.604	13.646	1439	160	7	2574	102
B-n39-k5	549	521.071	2.027	1407	86	6	—	—	—	—	—	—	—
B-n38-k6	805	741.634	0.924	1075	64	2	—	—	—	—	—	—	—
B-n35-k5	955	868.083	0.943	1176	70	3	926.018	345.966	3414	382	14	2353	136
B-n34-k5	788	755.231	0.988	1209	72	5	773.636	50.094	2179	220	11	2991	98
B-n31-k5	672	661.207	0.395	774	44	2	666.191	5.694	1207	121	6	1309	56
B-n57-k7	1153	1126.77	2.641	1881	99	4	—	—	—	—	—	—	—
B-n56-k7	707	635.601	3.472	1830	111	7	—	—	—	—	—	—	—
B-n52-k7	747	687.348	2.379	1601	95	5	—	—	—	—	—	—	—
B-n51-k7	1032	964.287	1.574	1351	74	4	—	—	—	—	—	—	—
B-n50-k8	1312	1266.45	1.629	1188	70	4	1282.79	39.079	1753	207	9	3178	111
B-n50-k7	741	690.528	2.067	1570	89	2	—	—	—	—	—	—	—
B-n45-k6	678	652.65	1.499	1043	65	4	666.834	50.453	1912	254	7	4386	130
B-n44-k7	909	865.181	0.712	991	55	2	880.067	568.004	2098	255	10	2892	131
B-n43-k6	742	703.674	1.029	1142	63	3	723.319	515.07	3149	353	15	3748	152
B-n78-k10	1221	1166.73	4.469	2689	141	3	—	—	—	—	—	—	—
B-n68-k9	1272	1198.2	3.112	2249	120	2	—	—	—	—	—	—	—

Continued on next page

Table V.3: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
B-n67-k10	1032	999.204	2.462	2113	111	2	1028.42	148.288	3526	319	12	6814	135
B-n66-k9	1316	1257.22	3.108	2077	111	4	—	—	—	—	—	—	—
B-n64-k9	861	808.253	2.762	1959	105	3	—	—	—	—	—	—	—
B-n57-k9	1598	1571.99	1.688	1550	85	3	1592.7	51.932	2707	219	7	2768	123
B-n63-k10	1496	1456.86	1.616	1634	88	2	1496	331.582	3381	294	11	3615	121
E-n76-k8	735	717.819	4.774	2680	140	2	732.929	297.473	4775	408	9	4048	195
E-n76-k7	682	664.202	8.118	3176	171	3	—	—	—	—	—	—	—
E-n76-k14	1021	1001.85	1.181	1288	67	2	1014.75	7.496	1627	145	4	1363	106
E-n76-k10	830	811.767	2.25	1903	98	1	825.787	28.129	2774	211	4	2535	160
E-n51-k5	521	517.135	1.971	1703	92	2	521	9.013	2008	128	5	500	32
E-n33-k4	835	820.918	3.131	1351	76	3	831.484	74.101	2611	212	5	1597	84
E-n30-k3	534	478.709	1.442	1339	89	3	—	—	—	—	—	—	—
E-n23-k3	569	564.413	5.998	944	68	2	569	12.521	977	87	2	706	8
E-n22-k4	375	373.875	0.093	357	23	1	375	0.112	361	25	1	86	3
E-n101-k8	815	789.372	16.902	4694	243	3	—	—	—	—	—	—	—
E-n101-k14	1067	1047.2	5.055	2889	147	2	1061.12	62.618	4028	291	6	3909	211
E-n31-k7	379	378	0.488	1015	59	1	379	0.613	1037	65	1	126	8
E-n13-k4	247	247	0.014	73	10	1	247	0.014	73	10	1	0	0
P-n55-k10	694	680.06	0.642	1023	55	1	689.193	5.284	1367	147	3	1224	100
P-n23-k8	529	529	0.016	563	4	1	529	0.018	563	4	1	0	0
P-n50-k10	696	687.347	0.49	812	45	2	694.223	1.42	969	84	2	433	53
P-n45-k5	510	502.349	1.257	1506	82	1	510	12.716	2234	145	4	1175	50
P-n40-k5	458	452.5	0.865	1294	68	1	458	4.621	1633	105	4	583	28

Continued on next page

Table V.3: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
P-n22-k8	603	601.25	0.078	146	11	1	603	0.033	155	16	2	16	3
P-n22-k2	216	214.5	0.585	883	64	2	216	1.415	1015	83	3	130	5
P-n21-k2	211	210.556	0.366	670	52	1	211	0.52	674	54	1	235	8
P-n20-k2	216	210	0.295	592	47	1	216	1.841	784	77	4	240	13
P-n19-k2	212	209.714	0.276	405	39	3	212	1.413	527	57	5	131	13
P-n16-k8	450	443.667	0.009	59	8	1	450	0.012	61	10	1	5	4
P-n101-k4	681	669.044	112.751	9909	577	9	—	—	—	—	—	—	—
P-n76-k5	627	614.86	16.081	4303	230	5	—	—	—	—	—	—	—
P-n76-k4	593	586.502	27.378	4929	286	7	593	948.867	7437	536	14	3500	73
P-n70-k10	827	809.29	1.531	1581	82	1	823.067	20.875	2358	196	4	2300	155
P-n65-k10	792	783.126	1.315	1334	70	2	792	5.993	1741	112	3	1338	89
P-n60-k15	968	959.606	0.34	784	40	1	967.465	1.169	898	75	2	504	55
P-n60-k10	744	736.528	1.051	1123	61	3	743.094	5.274	1577	119	3	1278	70
P-n55-k7	568	555.362	1.513	1518	80	1	564.192	45.879	2559	198	6	1929	118
P-n55-k15	989	968.361	7.48	162367	6	1	984.457	40.139	162450	38	2	378	53
P-n51-k10	741	732.955	0.382	830	43	1	740.778	1.989	1121	101	3	853	52
P-n50-k8	631	614.641	0.457	796	39	1	625.061	7.711	1249	125	4	1146	126
P-n50-k7	554	545.399	0.881	1222	63	1	554	7.828	1638	110	4	1063	63
M-n151-k12	1015	995.728	46.381	7398	381	4	1009.65	5094.99	11653	927	18	13943	349
M-n121-k7	1034	1026.37	61.983	6878	364	9	—	—	—	—	—	—	—
M-n101-k10	820	818.367	12.905	4644	251	12	820	75.75	6196	427	19	7434	15
M-n200-k16	1274	1250.23	133.6	9450	481	5	—	—	—	—	—	—	—

Continued on next page

Table V.3: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
M-n200-k17	1275	1252.52	134.934	9590	489	5	—	—	—	—	—	—	—
Average	839.271	820.654	2.592	3743.686	78.3	2.357	833.905	216.254	4535.757	190.043	6.171	2257.7	85

Table V.4: Capacity Cuts

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
A-n32-k5	784	784	1.536	1185	87	3	784	1.428	1185	87	3	0	0
A-n33-k5	661	661	0.555	823	56	2	661	0.668	823	57	2	78	1
A-n33-k6	742	740.25	0.971	904	68	3	742	1.065	916	72	3	27	5
A-n34-k5	778	774.429	1.244	1111	80	2	778	2.396	1205	98	3	101	14
A-n36-k5	799	798.322	2.728	1515	109	3	799	3.159	1520	113	3	323	8
A-n37-k5	669	666.625	2.299	1659	107	2	669	5.185	1915	135	3	89	10
A-n37-k6	949	937.027	1.03	942	65	3	945.577	9.546	1404	170	4	1077	58
A-n38-k5	730	723.4	2.078	1286	97	2	728.8	9.118	1544	153	6	665	54
A-n39-k5	822	817.524	4.207	1631	123	3	822	46.575	2432	230	5	1362	54
A-n39-k6	831	824.38	1.988	1365	98	2	831	13.268	1824	152	5	542	46
A-n44-k6	937	934.9	1.991	1258	92	2	937	2.331	1269	96	2	155	5
A-n45-k6	944	941.226	2.972	1548	112	4	944	8.115	1722	128	6	676	41
A-n45-k7	1146	1140.53	3.757	1515	122	2	1146	10.719	1769	156	5	660	56
A-n46-k7	914	914	2.33	1733	112	3	914	2.494	1762	115	2	138	1
A-n48-k7	1073	1071.54	2.981	1504	114	3	1073	10.765	1883	154	7	422	32
A-n53-k7	1010	1003.63	10.092	2615	206	2	1010	160.185	3196	301	8	2128	70
A-n54-k7	1167	1155.15	7.267	1986	148	4	1166.36	130.96	3173	338	9	2801	153
A-n55-k9	1073	1068.47	3.592	1683	116	3	1071.8	10.344	1988	173	3	1224	32
A-n60-k9	1354	1344.96	6.223	2167	144	2	1352.02	40.567	2865	237	5	2003	123
A-n61-k9	1034	1023.58	5.091	1745	126	3	1032	31.269	2242	220	6	1475	78
A-n62-k8	1288	1280.75	10.769	2622	179	4	1286.07	98.741	3433	346	10	4832	94
A-n63-k10	1314	1302.42	5.007	2073	133	5	1309.66	24.298	2665	216	5	1862	73

Continued on next page

Table V.4: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
A-n63-k9	1616	1608.54	9.112	2023	156	1	1616	44.395	2831	244	5	1482	65
A-n64-k9	1401	1387.16	5.764	2275	149	3	1395.86	124.498	3208	328	9	3491	125
A-n65-k9	1174	1166.56	7.007	2090	151	1	1174	35.977	2775	219	8	854	67
A-n69-k9	1159	1143.18	9.42	2463	165	2	1157.18	62.64	3370	276	5	2654	115
A-n80-k10	1763	1755.82	15.563	3434	215	4	1761.55	177.429	4614	396	9	3241	128
B-n45-k5	751	750.6	6.196	2566	173	6	751	7.888	2658	186	5	281	8
B-n41-k6	829	828.429	1.93	1246	98	3	829	2.46	1268	101	3	104	6
B-n39-k5	549	549	3.259	1862	133	4	549	3.249	1862	133	4	0	0
B-n38-k6	805	804.143	1.82	1383	104	3	805	2.418	1401	111	3	150	10
B-n35-k5	955	955	1.675	1713	109	2	955	1.724	1713	109	2	0	0
B-n34-k5	788	784.926	3.658	1897	148	2	788	13.108	2255	184	5	539	41
B-n31-k5	672	672	0.609	997	59	2	672	0.624	997	59	2	0	0
B-n57-k7	1153	1153	5.504	2352	148	4	1153	5.72	2352	148	4	0	0
B-n56-k7	707	705	6.278	2512	172	5	705	7.874	2548	181	5	483	4
B-n52-k7	747	746.333	6.075	2384	162	3	747	7.942	2447	169	3	738	10
B-n51-k7	1032	1026.83	6.75	2253	158	2	1032	535.377	2886	237	10	2438	82
B-n50-k8	1312	1303.39	4.563	1602	118	2	1309.35	42.724	2123	294	7	3259	68
B-n50-k7	741	741	4.917	2328	158	4	741	6.047	2368	164	4	619	3
B-n45-k6	678	677.973	2.684	1515	113	2	678	5.585	1619	127	3	500	22
B-n44-k7	909	909	1.963	1420	94	3	909	1.95	1420	94	3	0	0
B-n43-k6	742	736.825	2.601	1505	102	2	740.091	32.751	1978	219	5	2365	93
B-n78-k10	1221	1216.39	12.419	3348	209	2	1221	41.201	3803	264	7	2021	56
B-n68-k9	1272	1263.7	12.939	3006	202	3	1267.79	1060.51	4070	424	11	4180	140

Continued on next page

Table V.4: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
B-n67-k10	1032	1027.56	7.744	2634	179	4	1030.85	125.411	3416	333	10	3062	107
B-n66-k9	1316	1307.84	9.36	2769	183	2	1315.24	237.342	3788	378	10	3548	149
B-n64-k9	861	860.417	7.706	2779	176	3	861	9.044	2839	184	3	152	10
B-n57-k9	1598	1596	5.47	2061	150	4	1598	25.824	2405	218	7	2508	67
B-n63-k10	1496	1487.07	6.154	2166	150	3	1496	57.377	3241	262	8	1487	89
E-n76-k8	735	725.19	21.273	3307	233	2	734.091	469.267	5330	506	9	4543	187
E-n76-k7	682	669.015	11.097	3317	193	2	—	—	—	—	—	—	—
E-n76-k14	1021	1006.94	1.812	1362	79	2	1014.75	8.906	1662	148	3	1357	109
E-n76-k10	830	816.627	5.125	2040	130	2	826.44	46.967	2844	232	5	2469	156
E-n51-k5	521	518.357	4.529	1874	128	3	521	13.683	2202	160	4	500	37
E-n33-k4	835	835	4.343	1507	96	3	835	4.418	1507	96	3	0	0
E-n30-k3	534	518.1	3.572	1779	138	3	—	—	—	—	—	—	—
E-n23-k3	569	569	9.122	1023	91	3	569	9.195	1023	91	3	0	0
E-n22-k4	375	375	0.112	363	27	1	375	0.112	363	27	1	0	0
E-n101-k8	815	802.626	61.075	5594	364	4	—	—	—	—	—	—	—
E-n101-k14	1067	1053.52	8.969	3097	184	2	1063.29	87.762	4115	334	6	2739	193
E-n31-k7	379	378.333	0.795	1030	62	1	379	0.675	1054	70	1	46	5
E-n13-k4	247	247	0.021	73	10	1	247	0.014	73	10	1	0	0
P-n55-k10	694	681.754	1.086	1051	67	1	689.202	7.701	1330	144	4	1003	86
P-n23-k8	529	529	0.017	563	4	1	529	0.017	563	4	1	0	0
P-n50-k10	696	689.362	0.594	850	52	1	694.907	1.741	992	80	2	272	47
P-n45-k5	510	505.944	2.67	1628	110	1	510	16.43	2168	174	3	952	57
P-n40-k5	458	456.875	1.699	1395	87	1	458	5.474	1697	117	3	201	17

Continued on next page

Table V.4: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
P-n22-k8	603	603	0.034	147	14	1	603	0.028	147	14	1	0	0
P-n22-k2	216	215.5	0.77	884	68	2	216	1.157	971	80	1	60	3
P-n21-k2	211	211	0.464	672	54	1	211	0.477	672	54	1	0	0
P-n20-k2	216	213	0.526	650	60	2	216	1.264	719	83	2	198	9
P-n19-k2	212	212	0.369	443	44	2	212	0.36	443	44	2	0	0
P-n16-k8	450	448	0.033	64	12	1	450	0.024	64	14	1	1	1
P-n101-k4	681	676.428	392.77	12593	854	17	—	—	—	—	—	—	—
P-n76-k5	627	615.748	37.079	4666	308	4	—	—	—	—	—	—	—
P-n76-k4	593	587.459	62.493	5560	409	4	593	1533.34	8953	771	13	3500	76
P-n70-k10	827	813.335	2.631	1695	103	1	823.157	31.852	2353	222	4	2077	155
P-n65-k10	792	786.858	2.14	1432	90	2	792	5.785	1685	111	3	696	54
P-n60-k15	968	963.469	0.57	827	55	1	967.717	2.181	928	94	2	338	54
P-n60-k10	744	738.864	1.039	1188	68	1	743.513	6.475	1521	125	3	1013	73
P-n55-k7	568	558.912	2.467	1605	96	2	565.667	88.576	2597	216	6	1833	108
P-n55-k15	989	972.543	29.417	162386	18	2	984.614	210.101	162464	82	3	379	55
P-n51-k10	741	736.075	0.872	899	63	2	741	2.474	1042	86	3	357	43
P-n50-k8	631	616.66	0.723	851	52	1	625.753	10.61	1249	145	4	933	111
P-n50-k7	554	550.651	2.698	1431	96	2	554	6.103	1720	120	2	351	31
M-n151-k12	1015	998.665	73.574	7685	436	4	1012.1	4563.55	11909	968	19	9849	324
M-n121-k7	1034	1032.43	148.464	8962	543	11	1033.6	1544.63	10699	790	16	5993	42
M-n101-k10	820	820	20.81	5340	349	9	820	29.92	5721	378	11	1470	3
M-n200-k16	1274	1251.35	161.984	9784	528	5	—	—	—	—	—	—	—

Continued on next page

Table V.4: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
M-n200-k17	1275	1254.15	170.64	9854	540	4	—	—	—	—	—	—	—
Average	856.524	851.644	7.882	3801.631	126.405	2.607	855.595	143.019	4259.167	194.988	4.893	1261.024	53.679

Table V.5: DSSR + CB + CC

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
A-n32-k5	784	784	1.234	1180	84	1	784	1.257	1180	84	1	0	0
A-n33-k5	661	661	0.524	822	55	1	661	0.614	822	56	1	78	0
A-n33-k6	742	740.25	0.964	903	68	3	742	1.08	916	72	3	27	5
A-n34-k5	778	774.429	1.309	1111	80	2	778	2.423	1205	98	3	101	14
A-n36-k5	799	798.322	2.637	1513	109	3	799	3.54	1520	114	4	323	5
A-n37-k5	669	666.625	2.413	1659	107	2	669	5.327	1916	136	3	89	11
A-n37-k6	949	937.027	0.891	941	65	3	945.582	10.332	1401	171	5	1077	57
A-n38-k5	730	723.4	2.041	1284	97	2	728.8	9.654	1529	153	6	636	55
A-n39-k5	822	817.524	3.91	1631	123	3	822	76.514	2412	228	6	1362	61
A-n39-k6	831	824.395	2.082	1363	96	1	831	27.136	1820	154	7	542	47
A-n44-k6	937	934.9	2.085	1258	92	2	937	2.446	1269	96	2	155	5
A-n45-k6	944	941.238	2.992	1551	110	3	944	7.243	1705	123	3	676	47
A-n45-k7	1146	1140.53	4.692	1515	123	2	1146	10.894	1758	152	3	660	53
A-n46-k7	914	914	2.782	1747	114	2	914	2.955	1762	115	2	138	1
A-n48-k7	1073	1071.54	3.636	1504	114	3	1073	11.487	1874	148	5	422	37
A-n53-k7	1010	1003.63	12.283	2613	206	2	1010	244.663	3178	294	8	1899	69
A-n54-k7	1167	1155.17	7.223	1976	148	3	1166.38	196.794	3187	330	8	2867	150
A-n55-k9	1073	1068.47	3.322	1682	116	3	1071.8	10.571	1987	168	5	1168	43
A-n60-k9	1354	1344.96	5.334	2164	142	1	1352.04	61.802	2879	252	7	2164	121
A-n61-k9	1034	1023.58	4.642	1743	127	2	1032	41.279	2296	257	7	1706	79
A-n62-k8	1288	1280.75	9.813	2626	179	3	1286.07	123.024	3394	323	9	3846	93
A-n63-k10	1314	1302.43	4.058	2066	128	2	1309.67	25.685	2670	221	5	1953	74

Continued on next page

Table V.5: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
A-n63-k9	1616	1608.54	8.875	2023	156	1	1616	42.104	2833	248	5	1482	70
A-n64-k9	1401	1388.08	6.168	2274	153	3	1395.86	1098.3	3173	329	7	3356	134
A-n65-k9	1174	1166.56	6.657	2090	151	1	1174	30.899	2776	218	5	854	65
A-n69-k9	1159	1143.3	7.007	2459	165	2	1157.18	76.585	3353	278	4	2545	113
A-n80-k10	1763	1755.83	15.552	3426	217	4	1761.58	862.729	4580	395	7	2970	124
B-n45-k5	751	750.6	5.488	2562	168	4	751	7.335	2626	182	3	281	8
B-n41-k6	829	828.429	2.046	1246	98	3	829	2.257	1267	101	2	104	7
B-n39-k5	549	549	3.501	1860	137	4	549	3.463	1860	137	4	0	0
B-n38-k6	805	804.143	1.941	1383	104	3	805	2.54	1401	111	3	150	10
B-n35-k5	955	955	1.931	1713	109	2	955	1.942	1713	109	2	0	0
B-n34-k5	788	784.926	4.251	1897	148	2	788	15.622	2253	183	5	539	41
B-n31-k5	672	672	0.648	997	59	2	672	0.634	997	59	2	0	0
B-n57-k7	1153	1153	4.954	2336	146	3	1153	5.128	2336	146	3	0	0
B-n56-k7	707	705	5.711	2473	166	4	705	6.689	2484	169	3	483	3
B-n52-k7	747	746.333	5.825	2380	164	3	747	9.066	2452	175	4	859	11
B-n51-k7	1032	1026.83	6.218	2252	158	2	1032	3160.47	2880	234	9	2492	87
B-n50-k8	1312	1303.48	3.984	1591	117	2	1309.35	45.249	2186	308	7	3899	67
B-n50-k7	741	741	4.194	2319	158	4	741	4.806	2384	167	4	619	3
B-n45-k6	678	677.973	2.831	1515	113	2	678	5.035	1610	129	3	500	24
B-n44-k7	909	909	1.795	1417	94	2	909	1.734	1417	94	2	0	0
B-n43-k6	742	736.825	2.588	1503	101	1	740.09	119.843	1977	226	5	2311	89
B-n78-k10	1221	1216.39	12.983	3345	210	2	1221	37.655	3761	256	6	2000	50
B-n68-k9	1272	1263.71	11.017	2998	201	2	—	—	—	—	—	—	—

Continued on next page

Table V.5: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
B-n67-k10	1032	1027.59	6.717	2616	171	2	1030.86	655.782	3391	316	6	2581	120
B-n66-k9	1316	1307.84	8.963	2769	184	3	1315.24	3879	3791	384	10	3847	152
B-n64-k9	861	860.438	8.215	2777	176	3	861	9.305	2834	183	2	152	11
B-n57-k9	1598	1596	6.783	2065	152	4	1598	23.306	2389	216	4	2283	66
B-n63-k10	1496	1487.07	5.844	2162	150	3	1496	62.087	3207	257	5	1487	86
E-n76-k8	735	725.19	18.542	3254	227	2	734.092	1160.41	5323	525	9	4648	191
E-n76-k7	682	669.015	9.838	3317	193	2	—	—	—	—	—	—	—
E-n76-k14	1021	1006.94	1.559	1362	79	2	1014.75	10.203	1661	155	3	1371	106
E-n76-k10	830	816.627	5.075	2040	130	2	826.438	49.762	2844	238	5	2464	149
E-n51-k5	521	518.357	4.163	1874	128	3	521	13.395	2188	164	5	500	41
E-n33-k4	835	835	3.672	1514	99	3	835	3.687	1514	99	3	0	0
E-n30-k3	534	518.1	2.769	1779	138	3	—	—	—	—	—	—	—
E-n23-k3	569	569	7.326	1023	91	3	569	5.971	1023	91	3	0	0
E-n22-k4	375	375	0.098	363	27	1	375	0.1	363	27	1	0	0
E-n101-k8	815	802.626	63.515	5608	366	4	—	—	—	—	—	—	—
E-n101-k14	1067	1053.52	8.041	3097	184	2	1063.3	89.425	4106	334	6	2713	196
E-n31-k7	379	378.333	0.522	1030	62	1	379	0.646	1054	70	1	46	5
E-n13-k4	247	247	0.014	73	10	1	247	0.014	73	10	1	0	0
P-n55-k10	694	681.754	0.914	1051	67	1	689.203	7.707	1328	146	5	1013	86
P-n23-k8	529	529	0.022	563	4	1	529	0.03	563	4	1	0	0
P-n50-k10	696	689.362	0.589	850	52	1	695.019	1.817	992	80	2	250	47
P-n45-k5	510	505.944	2.585	1628	110	1	510	15.187	2168	174	3	952	57
P-n40-k5	458	456.875	1.631	1395	87	1	458	4.65	1694	116	3	201	17

Continued on next page

Table V.5: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
P-n22-k8	603	603	0.041	147	14	1	603	0.087	147	14	1	0	0
P-n22-k2	216	215.5	0.646	884	68	2	216	1.011	971	80	1	60	3
P-n21-k2	211	211	0.465	672	54	1	211	0.461	672	54	1	0	0
P-n20-k2	216	213	0.482	650	60	2	216	1.143	719	83	2	198	9
P-n19-k2	212	212	0.297	443	44	2	212	0.348	443	44	2	0	0
P-n16-k8	450	448	0.024	64	12	1	450	0.018	64	14	1	1	1
P-n101-k4	681	676.428	430.519	12573	867	14	—	—	—	—	—	—	—
P-n76-k5	627	615.748	38.204	4697	311	5	—	—	—	—	—	—	—
P-n76-k4	593	587.459	62.629	5560	409	4	593	2238.44	9002	763	15	3000	68
P-n70-k10	827	813.335	3.429	1695	103	1	823.172	45.677	2324	231	5	2082	157
P-n65-k10	792	786.858	1.886	1432	90	2	792	6.538	1682	111	3	696	54
P-n60-k15	968	963.469	0.578	827	55	1	967.717	2.004	928	94	2	338	54
P-n60-k10	744	738.864	1.033	1188	68	1	743.516	5.914	1512	118	2	975	73
P-n55-k7	568	558.922	1.879	1604	95	1	565.667	212.994	2603	218	5	1833	108
P-n55-k15	989	972.572	24.743	162385	16	1	985.347	140.187	162452	58	1	267	55
P-n51-k10	741	736.075	0.862	899	63	2	741	2.005	1038	85	2	357	44
P-n50-k8	631	616.66	0.666	851	52	1	625.773	12.586	1236	143	4	884	107
P-n50-k7	554	550.651	3.205	1430	95	1	554	6.153	1719	121	3	351	33
M-n151-k12	1015	998.727	83.733	7658	435	3	—	—	—	—	—	—	—
M-n121-k7	1034	1032.44	139.966	8777	532	1 0	1033.6	2476.88	10665	792	13	6820	49
M-n101-k10	820	820	20.708	5330	344	14	820	29.576	5806	383	13	970	4
M-n200-k16	1274	1251.35	206.593	9762	530	5	—	—	—	—	—	—	—

Continued on next page

Table V.5: *Continued from previous page*

Ins	OPT	without lm-SRC					with lm-SRC						
		Value	Time	Routes	Heu	Exa	Value	Time	Routes	Heu	Exa	Sep	Act
M-n200-k17	1275	1254.15	191.417	9812	534	3	—	—	—	—	—	—	—
Average	849.524	844.842	6.717	3759.232	121.207	2.259	848.672	214.528	4164.488	182.854	4.293	1105.768	49.780

Table V.6: Comparison with Pecin et al.

Ins	OPT	PPPU	Our results	Difference
A-n37-k5	669	669	669	0
A-n37-k6	949	946.02	945.582	0.438
A-n38-k5	730	730	728.8	1.2
A-n39-k5	822	822	822	0
A-n39-k6	831	831	831	0
A-n44-k6	937	937	937	0
A-n45-k6	944	944	944	0
A-n45-k7	1146	1146	1146	0
A-n46-k7	914	914	914	0
A-n48-k7	1073	1073	1073	0
A-n53-k7	1010	1010	1010	0
A-n54-k7	1167	1167	1166.38	0.62
A-n55-k9	1073	1073	1071.8	1.2
A-n60-k9	1354	1354	1352.04	1.96
A-n61-k9	1034	1034	1032	2
A-n62-k8	1288	1287.15	1286.07	1.08
A-n63-k9	1616	1616	1616	0
A-n63-k10	1314	1310.61	1309.67	0.94
A-n64-k9	1401	1394.55	1395.86	-1.31
A-n65-k9	1174	1174	1174	0
A-n69-k9	1159	1159	1157.18	1.82
A-n80-k10	1763	1763	1761.58	1.42
B-n38-k6	805	805	805	0
B-n39-k5	549	549	549	0
B-n41-k6	829	829	829	0
B-n43-k6	742	742	740.09	1.91
B-n44-k7	909	909	909	0
B-n45-k5	751	751	751	0
B-n45-k6	678	678	678	0
B-n50-k7	741	741	741	0
B-n50-k8	1312	1309.68	1309.35	0.33
B-n51-k7	1032	1032	1032	0
B-n52-k7	747	747	747	0
B-n56-k7	707	705	705	0
B-n57-k7	1153	1153	1153	0
B-n57-k9	1598	1598	1598	0
B-n63-k10	1496	1496	1496	0
B-n64-k9	861	861	861	0

Continued on next page

Table V.6: *Continued from previous page*

Ins	OPT	PPPU	Our results	Difference
B-n66-k9	1316	1316	1315.24	0.76
B-n67-k10	1032	1032	1030.86	1.14
B-n68-k9	1272	1267.45	—	—
B-n78-k10	1221	1221	1221	0
E-n51-k5	521	521	521	0
E-n76-k7	682	682	—	—
E-n76-k8	735	735	734.092	0.908
E-n76-k10	830	828.2	826.438	1.762
E-n76-k14	1021	1016.24	1014.75	1.49
E-n101-k8	815	815	—	—
E-n101-k14	1067	1063.91	1063.3	0.61
M-n101-k10	820	820	820	0
M-n121-k7	1034	1034	1033.6	0.4
M-n151-k12	1015	1011.74	—	—
M-n200-k16	1274	1266.53	—	—
M-n200-k17	1275	1268.71	—	—
P-n16-k8	450	448	450	-2
P-n19-k2	212	212	212	0
P-n20-k2	216	216	216	0
P-n21-k2	211	211	211	0
P-n22-k2	216	216	216	0
P-n22-k8	603	603	603	0
P-n23-k8	529	529	529	0
P-n40-k5	458	458	458	0
P-n45-k5	510	510	510	0
P-n50-k7	554	554	554	0
P-n50-k8	631	628.65	625.773	2.877
P-n50-k10	696	696	695.019	0.981
P-n51-k10	741	741	741	0
P-n55-k7	568	566.02	565.667	0.353
P-n55-k10	694	690.19	689.203	0.987
P-n55-k15	989	987.02	985.347	1.673
P-n60-k10	744	743.68	743.516	0.164
P-n60-k15	968	963.47	967.717	-4.247
P-n65-k10	792	792	792	0
P-n70-k10	827	823.89	823.172	0.718
P-n76-k4	593	593	593	0
P-n76-k5	627	627	—	—

Continued on next page

Table V.6: *Continued from previous page*

Ins	OPT	PPPU	Our results	Difference
P-n101-k4	681	681	—	—
Average	885.174	884.482	884.161	0.322

VI
Conclusion

This master thesis proposed to assess the leverage achieved with the combination of the lm-SRC with other techniques, like Decremental Space State Relaxation (DSSR), Completion Bounds, *ng*-routes and capacity cuts over a Set Partitioning Formulation for the problem.

Observing the results presented in Chapter V we can highlight some points:

- The addition of lm-SRC was responsible to improve the bounds for almost all instances with running time smaller than two hours however when combining them with capacity cuts, the bounds have a great improvement. The average difference between the optimum value and the bound found decreased from 5.0 to 0.9 in the last two configurations.
- The column generation lower bounds found were on majority equal to the ones found by Pecin et al. [19]. When they were better, the difference was quite small and for 3 instances our lower bounds were better. However, 9 instances found in their work did not finished within the time limit.
- The time required by the lm-SRC did not improved with the consideration of DSSR and Completion Bounds when using these techniques with respect to the *ng*-routes. These techniques can be adapted to the lm-SRC, but it was not implemented in this work.

As future works and extensions we suggest: (*i*) a route enumeration to verify which instances can be solved; and (*ii*) the improvement of the approach started in this work. Techniques that speed up the pricing as reduced cost variable fixing, ways to strengthen the completion bounds should be tested, as well as other strong cuts which can be separated in the SP formulation (other lm-SRCs and Strong Degree cuts, for example).

Bibliography

[1] DANTZIG, G. B.; RAMSER, J. H.. **The truck dispatching problem.** Management science, 1959. I

[2] TOTH, P.; VIGO, D.. **The Vehicle Routing Problem.** Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia, 2002. I

[3] FORD JR, L. R.; FULKERSON, D. R.. **A suggested computation for maximal multi-commodity network flows.** Management Science, 5(1):97–101, 1958. I

[4] LAPORTE, G.; NOBERT, Y.. **A branch and bound algorithm for the capacitated vehicle routing problem.** OR Spectrum, 1983. I, II

[5] BALINSKI, M. L.; QUANDT, R. E.. **On an integer program for a delivery problem.** Operations Research, 1964. I, II

[6] DROR, M.. **Note on the complexity of the shortest path models for column generation in vrptw.** Operations Research, 42(5):977–978, 1994. I

[7] CHRISTOFIDES, N.; MINGOZZI, A.; TOTH, P.. **Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations.** Mathematical programming, 1981. I, III.1

[8] IRNICH, S.; VILLENEUVE, D.. **The Shortest-Path Problem with Resource Constraints and k-Cycle Elimination for $k \geq 3$.** INFORMS Journal on Computing, 18(3):391–406, 2006. I, I.1

[9] BALDACCI, R.; MINGOZZI, A.; ROBERTI, R.. **New route relaxation and pricing strategies for the vehicle routing problem.** Operations Research, 59(5):1269–1283, 2011. I, I.1, III.2, IV.2

[10] FUKASAWA, R.; LONGO, H.; LYSGAARD, J.; POGGI DE ARAGÃO, M.; REIS, M.; UCHOA, E.; WERNECK, R. F.. **Robust Branch-and-Cut-and-Price for the Capacitated Vehicle Routing Problem.** Mathematical Programming, 106(3):491–511, 2006. I.1, III.1

[11] BALDACCI, R.; CHRISTOFIDES, N.; MINGOZZI, A.. **An Exact Algorithm for the Vehicle Routing Problem Based on the Set Partitioning**

Formulation with Additional Cuts. Math. Program., 115(2):351–385, 2008. I.1

[12] PESSOA, A.; DE ARAGÃO, M. P.; UCHOA, E.. **Robust branch-cut-and-price algorithms for vehicle routing problems**. In: THE VEHICLE ROUTING PROBLEM: LATEST ADVANCES AND NEW CHALLENGES, p. 297–325. Springer, 2008. I.1

[13] MARTINELLI, R.; PECIN, D.; POGGI, M.; LONGO, H.. **A branch-cut-and-price algorithm for the capacitated arc routing problem**. In: Pardalos, P.; Rebennack, S., editors, EXPERIMENTAL ALGORITHMS, v. 6630 of **Lecture Notes in Computer Science**, p. 315–326. Springer Berlin / Heidelberg, 2011. I.1

[14] RIGHINI, G.; SALANI, M.. **Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints**. Discrete Optimization, 3(3):255–273, Sept. 2006. I.1

[15] MARTINELLI, R.; PECIN, D.; POGGI, M.. **Efficient elementary and restricted non-elementary route pricing**. European Journal of Operational Research, 2014. I.1, III.2

[16] CONTARDO, C.. **A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints**. Technical report, Archipel-UQAM 5078, Université du Québec à Montréal, Canada, 2012. I.1

[17] RØPKE, S.. **Branching decisions in branch-and-cut-and-price algorithms for vehicle routing problems**. Presentation in Column Generation, 2012. I.1

[18] CONTARDO, C.; MARTINELLI, R.. **A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints**. Discrete Optimization, 12:129–146, 2014. I.1, IV.2

[19] PECIN, D.; PESSOA, A.; POGGI, M.; UCHOA, E.. **Improved branch-cut-and-price for capacitated vehicle routing**. In: INTEGER PROGRAMMING AND COMBINATORIAL OPTIMIZATION, p. 393–403. Springer, 2014. I.1, IV.3, V, VI

[20] JEPSEN, M.; PETERSEN, B.; SPOORENDONK, S.; PISINGER, D.. **Subset-row inequalities applied to the vehicle-routing problem with time windows**. Operations Research, 2008. I.1, IV, IV.2

[21] PECIN, D.; PESSOA, A.; POGGI, M.; UCHOA, E.. **Improved branch-cut-and-price for capacitated vehicle routing**. In: INTEGER PROGRAMMING AND COMBINATORIAL OPTIMIZATION. Springer, 2014. I.2

[22] GAREY, M. R.; JOHNSON, D. S.. **Computers and Intractability; A Guide to the Theory of NP-Completeness**. W. H. Freeman & Co., New York, NY, USA, 1979. II.1, IV.1

[23] RIGHINI, G.; SALANI, M.. **New Dynamic Programming Algorithms for the Resource Constrained Elementary Shortest Path Problem**. Networks, 51(3):155–170, 2008. III.2

[24] LYSGAARD, J.; LETCHFORD, A. N.; EGLESE, R. W.. **A new branch-and-cut algorithm for the capacitated vehicle routing problem**. Mathematical Programming, 100(2):423–445, June 2004. IV

[25] LYSGAARD, J.. **CVRPSEP: A package of separation routines for the capacitated vehicle routing problem**. Institut for Driftøkonomi og Logistik, Handelshøjskolen i Århus, 2003. IV.1

[26] GUROBI OPTIMIZATION, I.. **Gurobi optimizer reference manual**, 2015. V